P. 1877/85

mikroKLAN 21

9 1985

informatyka

Złożoność algorytmów

Rozległe sieci komputerowe

Jeszcze o LOGO

Komputer dydaktyczny

Cena 100 zł

informatyka

9 1985

Nr 9

Miesięcznik Rok XX

Wrzesień 1985

Organ Komitetu Informatyki MNSZWiT oraz Komitetu Naukowo-Technicznego NOT ds. Informatyki

KOLEGIUM REDAKCYJNE:

Dr inż. Wacław ISZKOWSKI, mgr Teresa JABŁOŃSKA (sekretarz redakcji), Władysław KLEPACZ (zastępca redaktora naczelnego), prof. dr hab. Leon ŁUKASZEWICZ (redaktor naczelny), mgr inż. Andrzej J. PIOTROWSKI, dr inż. Janusz ZALEWSKI

STALE WSPÓŁPRACUJĄ:

Mgr inż. Witold ABRAMOWICZ (Szwajcaria), mgr inż. Ryszard K. KOTT (Wielka Brytania), mgr inż. Teresa WILCZEK

PRZEWODNICZĄCY RADY PROGRAMOWEJ:

Prof. dr hab. Juliusz Lech Kulikowski

Materiałów nie zamówionych redakcja nie zwraca

Redakcja: 00-041 Warszawa, ul. Jasna 14/16, pok. 243 i 244, tel. 27-71-40 lub 26-82-61 w. 184

Zakł. Graf. „Tamka". Zam. 0761-1300/85. Obj. 4,0 ark. druk. Nakład 6300 egz. P-78.

ISSN 0542-9951, INDEKS 36124

Cena egzemplarza 100 zł
Prenumerata roczna 1200 zł

WYDAWNICTWO

NACZELNA ORGANIZACJA TECHNICZNA

CZASOPISM I KSIĄŻEK TECHNICZNYCH

SIGMA

00-950 Warszawa
skrytka pocztowa 1004
ul. Biała 4

W NUMERZE

	Strona
Złożoność algorytmów (1). Klasyfikacja problemów *Bolesław Mikołajczak, Janusz Stokłosa*	1
Rozległe sieci komputerowe *Eugeniusz Bilski, Zbigniew Fryźlewicz, Zbigniew Huzar, Andrzej Kaliś*	4
LOGO (3). Porównanie dialektów APPLE, SINCLAIR i COMMODORE 64 *Stanisław Waligórski*	12
EW — elastycznie mikroprogramowany komputer dydaktyczny *Adam Gacek, Stanisław Kozielski, Piotr Stróżyna*	22
SAMOTESTY	26
IV/A. Metodyka programowania	
mikroKLAN	13
Mikrokomputerowa pierwsza pomoc	
Wyścig z czasem	
Sposoby kreślenia okręgów	
Przykłady procedur w języku ASSEMBLER 8080	
Zapamiętywanie obrazu w CPC 464	
TERMINOLOGIA	27
Nazwy własne produktów informatycznych	
ZE ŚWIATA	30

W NASTĘPNYCH NUMERACH:

- Stanisław Gasik o wprowadzeniu do programometrii
- Zdzisław Płoski o RATFORZE
- Krzysztof Perycz o egzekutorze E6MR dla R-32
- Zbigniew Szkaradnik porównuje języki programowania
- Wiesław Nosowski i Bolesław Szomański o pakiecie symulacyjnym
- Jan Bielecki o programowaniu asemblerowym w języku fig-FORTH
- Jerzy Sukiennik o rozrachunku gospodarczym zakładowego ośrodka informatyki
- Edward Bieleninik i inni o podsieci komunikacyjnej Międzyuczelnianej Sieci Komputerowej

BOLESŁAW MIKOŁAJCZAK
JANUSZ STOKŁOSA
Instytut Automatyki
Politechnika Poznańska

P. 1877/85

Złożoność algorytmów (1)

Klasyfikacja problemów

W informatyce pod pojęciem algorytmu intuicyjnie rozumie się szczegółowy przepis określający jak powinien przebiegać proces obliczeń. Zwykle z każdym algorytmem związane są trzy zasadnicze pytania: czy dla rozpatrywanego zadania algorytm istnieje, jeśli tak, to jak go skonstruować i jaka jest jego złożoność?

Część pierwsza artykułu zawiera wprowadzenie w zagadnienia przedstawione w tytule, klasyfikację problemów ze względu na stopień trudności obliczeń, omówienie złożoności opisu struktury algorytmów oraz analizę złożoności obliczeń programów w języku asemblera. W drugiej części zawarte będą rozważania o asymptotycznej złożoności algorytmów oraz o problemach trudnych i łatwych obliczeniowo.

EFEKTYWNOŚĆ OBLICZEŃ

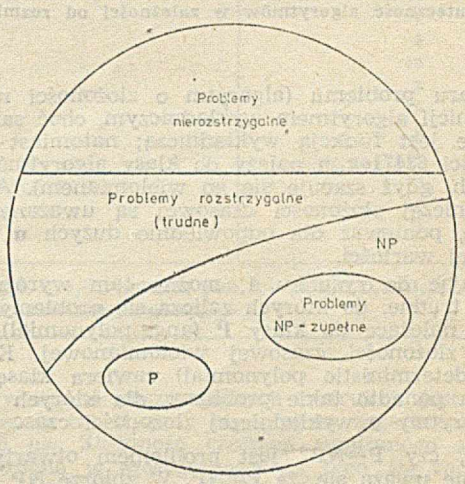
W latach trzydziestych naszego stulecia sformalizowano pojęcie obliczalności efektywnej, ale dopiero w latach siedemdziesiątych zaczęto się głębiej interesować tym, co jest łatwo, a co trudno obliczalne. **Łatwo** znaczy tutaj **praktycznie.**

Wyobraźmy sobie sytuację, w której agent handlowy musi odwiedzić pewną liczbę miast, każde jeden raz i wrócić do miejsca rozpoczęcia podróży, przebywszy najmniejszą liczbę kilometrów (zakładamy, że odległości między miastami są znane). Jeśli liczba miast będzie niewielka (np. 4 lub 5), można pokusić się o rozwiązanie tego problemu za pomocą kartki i ołówka, gdy będzie większa (np. kilkanaście lub kilkadziesiąt) narzędzie trzeba zmienić — kartkę i ołówek spróbujemy zastąpić komputerem. Do rozwiązania problemu skonstruujemy algorytm, który generuje wszystkie możliwe drogi, oblicza odległości i wybiera najkrótszą z nich. Jeśli agent ma odwiedzić **n** miast, to liczba wszystkich możliwych dróg wynosi $\frac{n!}{2}$ (kierunek jazdy nie odgrywa roli). Przyjmijmy, że dysponujemy komputerem, który na wygenerowanie każdej możliwej trasy zużywa 10^{-9} **s**, a liczba miast wynosi 23. Przy tych założeniach nasz komputer na wygenerowanie wszystkich możliwych marszrut musiałby zużyć ponad 10^{14} lat, a to raczej zniechęca (dla porównania: wiek Ziemi szacuje się na ok. $4{,}6 \cdot 10^9$ lat). Użycie szybszego komputera niewiele zmieni sytuację.

Przedstawiony problem, zwany problemem komiwojażera, możemy zaliczyć zatem do problemów trudnych.

Nie oznacza to, jak widzieliśmy, że pewne przypadki szczególne tego problemu mogą być rozwiązane małym nakładem kosztów, lecz w ogólności (dla dowolnej liczby miast) tak nie jest [1]).

Przed wprowadzeniem klasyfikacji problemów podamy kilka określeń. **Problemem** będziemy nazywać pytanie ogólne, zależne zwykle od kilku parametrów, na które należy znaleźć odpowiedź. Problem uważa się za zadany, gdy przedstawiony jest opis jego parametrów (w problemie komiwojażera były to odległości między miastami) i zdanie stwierdzające, czym winna charakteryzować się odpowiedź, czyli rozwiązanie. Z przypadkiem szczególnym problemu mamy do czynienia wówczas, gdy ustalimy wartości wszystkich jego parametrów.



Rys. 1. Klasyfikacja problemów (przy założeniu, że NP ≠ P)

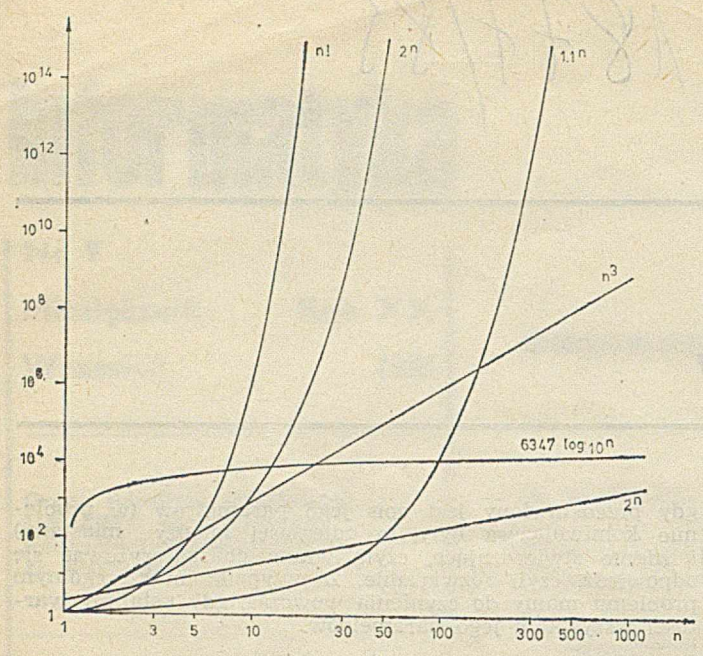
Klasyfikację problemów przedstawiono na rysunku 1. Problemy dzielą się na rozstrzygalne i nierozstrzygalne. **Problemy nierozstrzygalne** to takie, dla których nie istnieją algorytmy, które je rozwiązują. W tej sytuacji rozwiązanie każdego przypadku lub grupy przypadków wymaga specyficznej metody, gdyż nie istnieje jedna metoda rozwiązania problemu. W niniejszym artykule będziemy się zajmować takimi problemami, które można rozwiązać za pomocą algorytmów, czyli **problemami rozstrzygalnymi.**

Niech **f(n)** i **g(n)** będą funkcjami określonymi w zbiorze liczb naturalnych. Mówimy, że **f(n)** jest rzędu **g(n)** i zapisujemy to w postaci $f(n)=0(g(n))$, jeżeli istnieje liczba naturalna **k**, taka że $f(n) \leqslant k.g(n)$ dla prawie wszystkich **n** (czyli dla wszystkich z wyjątkiem pewnego skończonego, być może pustego, zbioru wartości **n**). **Złożonością czasową algorytmu** nazywamy czas potrzebny do jego wykonania, wyrażony w funkcji rozmiaru problemu (rozmiarem problemu może być stopień macierzy kwadratowej — w problemie dotyczącym macierzy kwadratowych, liczba wierzchołków grafu — w problemie dotyczącym grafu itd.). Rozmiar pamięci urządzenia liczącego potrzebny do wykonania algorytmu, wyrażony w funkcji rozmiaru problemu, nazywa się **złożonością pamięciową algorytmu.**

Algorytmem o złożoności czasowej (odp. pamięciowej) **wielomianowej** nazywamy algorytm, którego złożoność czasowa (odp. pamięciowa) jest $0(p(n))$, gdzie $p(n)$ jest wielomianem, natomiast **n** oznacza rozmiar problemu. W przypadku przeciwnym algorytm nazywa się **algorytmem o wykładniczej** — lub dokładniej: co najmniej wykładniczej — **złożoności czasowej** (odp. pamięciowej).

Wydajność algorytmu zależy od jego złożoności w funkcji rozmiaru problemu. Na rysunku 2 zilustrowano skuteczność algorytmów o różnych złożonościach, w zależności

[1]) Sytuacji nie zmienia fakt, że zastosowanie innego algorytmu, opartego na programowaniu liniowym, pozwoliło rozwiązać problem komiwojażera dla 318 miast [2]. Jest to jak dotąd (por. [7]) największy rozwiązany przypadek, dla którego udowodniono poprawność rozwiązania.



Rys. 2. Skuteczność algorytmów w zależności od rozmiarów problemów

od rozmiaru problemu (algorytm o złożoności **n!** jest w myśl definicji algorytmem wykładniczym, choć sama funkcja **n!** nie jest funkcją wykładniczą; natomiast algorytm o złożoności $63471\log_{10}n$ należy do klasy algorytmów wielomianowych, gdyż szacuje się go wielomianem). Algorytmy o wykładniczej złożoności czasowej są uważane za nieefektywne, ponieważ dla odpowiednio dużych **n** nie mają praktycznej wartości.

Powracając do rysunku 1, można tam wyróżnić także problemy trudne, do których zalicza się problemy **P** i **NP**. Problemy należące do klasy **P** (ang. polynomial), to problemy o złożoności czasowej wielomianowej. Klasa **NP** (ang. nondeterministic polynomial) zawiera klasę **P** problemów, a ponadto takie problemy, dla których znane są tylko algorytmy o wykładniczej złożoności czasowej [2]).

Problem, czy **P=NP?** jest problemem otwartym, choć powszechnie uważa się, że **P≠NP.** W zbiorze **NP** wyróżnia się podzbiór problemów **NP-zupełnych,** charakteryzujących się ciekawą własnością — wszystkie problemy z klasy **NP** można efektywnie zredukować do problemów **NP-zupełnych** [3]). Wynika stąd, że jeśli choć jeden z problemów **NP-zupełnych** dałoby się rozwiązać za pomocą algorytmu o złożoności wielomianowej, to każdy problem z klasy **NP** można by rozwiązać za pomocą tego typu algorytmu [4, 9].

ZŁOŻONOŚĆ OPISU ALGORYTMU

Wszystko, co dotąd powiedziano o złożoności algorytmów dotyczyło ich złożoności obliczeniowej. Oprócz tego można mówić także o złożoności opisu (struktury) algorytmu. **Złożoność opisu** jest to liczba charakteryzująca stopień komplikacji opisu algorytmu. W sposób pośredni jest związana ze złożonością obliczeń, choć bezpośrednio, jak się wydaje, większe znaczenie ma przy uruchamianiu programów [8]. Będziemy przyjmować, że algorytmy są przedstawione za pomocą sieci działań (schematu blokowego), lub że są zapisane w pewnym języku programowania.

Złożoność opisu może być rozumiana jako stopień komplikacji sieci działań algorytmu. Za miarę złożoności można by przyjąć liczbę możliwych przejść przez sieć działań. Jednakże w każdym programie zawierającym pętle istnieje potencjalnie nieskończona liczba przejść [10]. Miara złożoności opisu będzie oparta na pojęciu liczby cyklomatycznej grafu, skonstruowanego dla danej sieci działań [8]. (Liczba cyklomatyczna grafu **G** złożonego z **n** wierzchołków, **e** krawędzi oraz **p** spójnych komponentów wynosi $v(G)=e-n+p$ i jest liczbą nieujemną).

Dla każdej sieci działań algorytmu można skonstruować graf skierowany **G** (zwany grafem sterowania) z jednym wejściem i jednym wyjściem w taki sposób, że wierzchołek grafu odpowiada sekwencji instrukcji programu wykonywanej szeregowo, a krawędzie grafu odpowiadają rozgałęzieniom w programie. Zakłada się przy tym, że z każdego wierzchołka grafu można osiągnąć wierzchołek wyjściowy.

Złożoność cyklomatyczna grafu G sterowania jest to liczba

$$c(G) = e - n + a,$$

gdzie

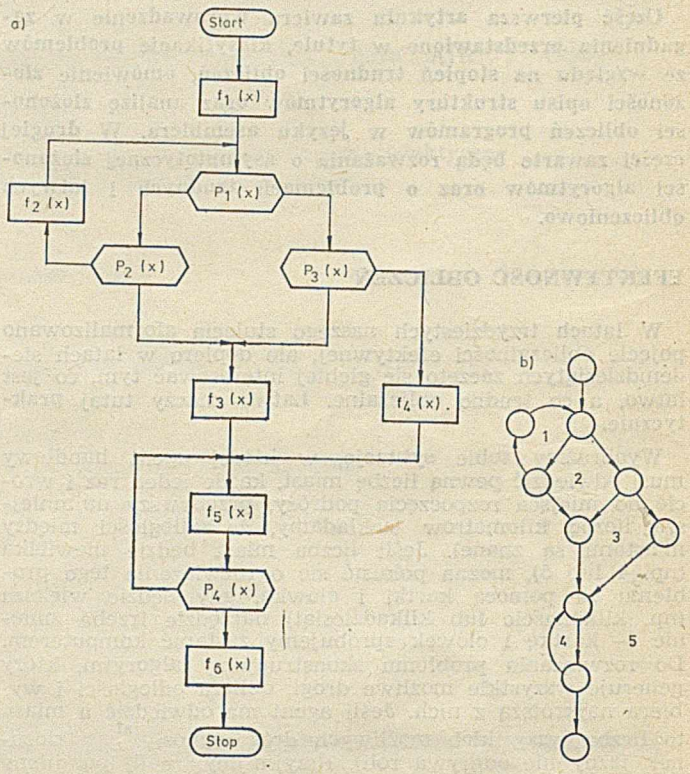
$$a = \begin{cases} 1 & \text{gdy } G \text{ jest silnie spójny} \\ 2 & \text{w przypadku przeciwnym} \end{cases}$$

W przypadku spójnego grafu planarnego o r reginach [4])

$$c(G) = r$$

Dla przykładu rozpatrzmy sieć działań przedstawioną na rysunku 3a. Graf sterowania dla tej sieci pokazano na rysunku 3b. W tym przypadku **e=13, n=10, r=5.** Wobec tego

$$c(G) = e - n + 2 = r = 5$$



Rys. 3. a) Sieć działań algorytmu,
b) graf sterowania dla tej sieci

Z tego co powiedziano wynika, że złożoność cyklomatyczna nie zależy od fizycznych rozmiarów programu, lecz wyłącznie od jego struktury decyzyjnej.

Zdefiniujemy teraz miarę złożoności dla sieci działań modułów programów [6]. Moduł jest rozumiany jako zespół jednostek programów wydzielonych z całości struktury i korzystający z wydzielonych danych. Dane te są używane wspólnie przez wszystkie programy modułu, są niedostępne dla innych programów i są chronione pomiędzy kolejnymi wywołaniami modułu. Inna jest tutaj interpretacja grafu sterowania: każdemu wierzchołkowi odpowiada pojedynczy moduł, a krawędzie reprezentują powiązania międzymodułowe (wywołanie i powrót sterowania). Będziemy przyjmować, że miara złożoności C_M sieci modułów — przy założeniu, że $C_1, C_2, \ldots C_k$ są złożonościami poszczególnych modułów, a C_N jest złożonością grafu sterowania sieci — wynosi

[2]) Precyzyjniej mówiąc, do klasy NP należą problemy rozwiązywalne za pomocą niedeterministycznej maszyny Turinga w czasie wielomianowym, natomiast klasa P to zbiór problemów rozwiązywalnych za pomocą deterministycznej maszyny Turinga w czasie wielomianowym.

[3]) Efektywna redukowalność oznacza przekształcenie jednego problemu w inny, za pomocą deterministycznej maszyny Turinga o wielomianowej złożoności czasowej (por. [1], [9]).

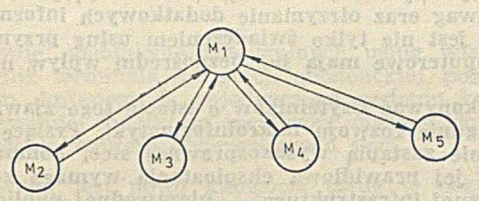
[4]) Regionami nazywamy obszary, na jakie geometryczna reprezentacja grafu planarnego dzieli płaszczyznę [3].

$$C_M = w_1 C_N + w_2 \sum_{i=1}^{k} C_i$$

gdzie w_1, w_2 są współczynnikami wagowymi.

Jako przykład rozpatrzmy sieć przedstawioną na rysunku 4, składającą się z monitora M_1 i zarządzanych przez niego modułów M_2, M_3, M_4 i M_5. Zakładając, że $C_N = c(G)$ otrzymujemy

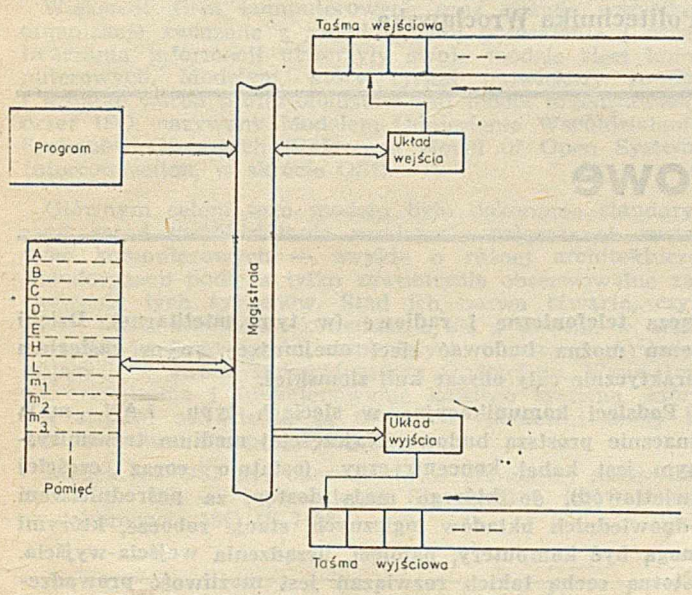
$$C_M = 4w_1 + w_2 \sum_{i=1}^{5} C_i$$



Rys. 4. Graf sterowania sieci modułów

ZŁOŻONOŚĆ OBLICZEŃ PROGRAMÓW W JĘZYKU ASEMBLERA

W poniższych rozważaniach będziemy posługiwać się modelem komputera przedstawionym na rysunku 5 (komórki **A, B, C, D, E, H, L** pamięci są komórkami wyróżnionymi, a taśmy wejściowa i wyjściowa mają numery **0**). Będziemy przyjmować, że każdy program składa się z ciągu instrukcji należących do zbioru instrukcji: MOV, LDA, STA, JMP, JZ, JNZ, INR, DCR, ADD, SUB, IN, OUT, HLT i że ich znaczenie, jest takie samo jak w języku asemblera INTEL 8080 [5]. Zakładamy, że przed wykonywaniem programu wszystkie komórki pamięci są automatycznie zerowane, każda z nich jest 8-bitowa i że ich liczba jest skończona, lecz praktycznie wystarczająco duża do wykonania wszystkich zakładanych obliczeń. Oznacza to, że taki model możemy stosować, jeśli liczby naturalne, pojawiające się w trakcie obliczeń, nie przekroczą pewnego górnego zakresu i zadanie można rozwiązać korzystając z pamięci operacyjnej komputera.



Rys. 5. Schemat funkcjonalny uproszczonego komputera

W charakterze przykładu napiszemy program służący do obliczania wartości funkcji $f(x,y,z) = xy + z$. Dane w postaci heksadecymalnej są umieszczone na taśmie wejściowej w taki sposób, że w pierwszej komórce taśmy znajduje się **x**, w drugiej **y**, natomiast w trzeciej **z**. Liczba **x** jest przechowywana w komórce **B**, **y** w komórce **C**, a liczba **z** w komórce **D**. Mnożenie jest wykonywane poprzez (y—1)-krotne dodanie liczby **x** do siebie. Do przechowywania wyników częściowych służy komórka **E**.
Niech program przyjmie postać:

```
      IN 0
      MOV B,A          ; x w B
      MOV E,A          ; oraz w E
      IN 0
      MOV C,A
      DCR C            ; y—1 w C
      IN 0
      MOV D,A          ; z w D
LAB:  MOV A,E
      ADD B
      MOV E,A          ; zawartość E powiększona o x
      DCR C
      JNZ LAB          ; jeśli dodawania nie wykonano
                       ; (y—1)-krotnie, skocz do LAB
      MOV A,E
      ADD D
      OUT 0            ; wynik na taśmie wyjściowej
      HLT
```

Czasy wykonywania instrukcji

Instrukcja[1])	Czas wykonywania[2])
MOV r_1, r_2	5
MOV M, r	7
MOV r, M	7
LDA	13
STA	13
JMP	10
JZ	10
JNZ	10
INR r	5
INR M	10
DCR r	5
DCR M	10
ADD r_2	4
SUB r_2	4
IN	10
OUT	10
HLT	7

[1]) $r_1 \neq M \neq r_2$, r, r_1, $r_2 \in R$, $r_2 \in R \cup \{M\}$ gdzie $R = \{A, B, C, D, E, H, L\}$
[2]) wyrażony w okresach zegara taktującego

Koszt wykonania poszczególnych instrukcji jest taki jak to pokazano w tabeli, przy czym okres zegara taktującego wynosi 0,5 μs. Złożoność czasowa napisanego programu jest uzależniona w szczególności od tego, ile razy będzie wykonywana pętla, a więc zależy bezpośrednio od **y**. Po szczegółowych obliczeniach okazuje się, że wynosi ona $(29y + 52)$ cykli, czyli $(14{,}5y + 26) \cdot 10^{-6}$ s.

Przyjęte założenia, w szczególności dotyczące 8-bitowej reprezentacji liczb, mogą sugerować, że zakres zastosowań omawianego modelu jest ograniczony. Przypuszczenie takie okaże się jednak fałszywe, jeśli uświadomić sobie, że można wykonywać obliczenia wielokrotnej precyzji. Zastosowanie operacji wielokrotnej precyzji powoduje jednakże wydłużenie czasu obliczeń.

LITERATURA

[1] Aho A. V., Hopcroft J. E., Ullman J. D.: Projektowanie i analiza algorytmów komputerowych. PWN, Warszawa, 1983
[2] Crowder H., Padberg M. W.: Solving large-scale symmetric traveling salesman problems to optimality. Management Science, vol. 26, 1980, 495—509
[3] Deo N.: Teoria grafów i jej zastosowania w technice i informatyce. PWN, Warszawa, 1980
[4] Garey M. R., Johnson D. S.: Computers and Intractability. A Guide to the Theory of NP-Completness. W. H. Freeman, San Francisco, 1979
[5] Grabowski J., Koślacz S.: Podstawy i praktyka programowania mikroprocesorów. WNT, Warszawa, 1980
[6] Hall N. R., Preiser S.: Combined network complexity measures. IBM Journal of Research and Development, vol. 28, 1984, 15—27
[7] Held M., Hoffman A. J., Johnson E. L., Wolfe P.: Aspects of the traveling salesman problem. IBM Journal of Research and Development, vol. 28, 1984, 476—486
[8] McCabe T. J.: A complexity measure. IEEE Transactions on Software Engineering, vol. SE-2, 1976, 308—320
[9] Mikołajczak B., Stokłosa J.: Złożoność obliczeniowa algorytmów. Wydawnictwo Politechniki Poznańskiej, Poznań 1984
[10] Varga L.: On software complexity. W.: Arató M., Varga L. (eds.): Mathematical Models in Computer Systems. Akadémiai Kiadó, Budapest, 1981, 183—197.

Kilka lat temu, kolega po powrocie ze Szwecji zaprosił nas na oglądanie slajdów. Na ekranie, na kolejnych obrazkach, na stole w nowoczesnym wnętrzu pokoju stała Ona — o miłych dla oka kształtach — ze słuchawką telefoniczną. Wszystkie slajdy były poświęcone tylko Jej, a kolega prawie z wypiekami na twarzy opowiadał, ile to wieczorów i nocy przy niej spędził, nie ruszając się na krok z domu. Była to końcówka komputera przyłączonego do sieci — prawie cały świat stał się przez to dostępny.

Sieci komputerowe stały się już faktem. Bez nich nie można sobie wyobrazić nowoczesnych usług bankowych, administrowania przedsiębiorstwami, zarządzania komunikacją, prowadzenia rozległych badań naukowych oraz szybkiej wymiany informacji. Dla przykładu, w wielu czasopismach naukowych (np. CACM) oprócz adresu autora znajduje się hasło poczty komputerowej, umożliwiające czytelnikowi bezpośrednie przesłanie swoich uwag oraz otrzymanie dodatkowych informacji.

Posiadanie sieci komputerowej jest nie tylko świadczeniem usług przynoszących zysk jej właścicielowi. Sieci komputerowe mają już bezpośredni wpływ na wzrost dochodu narodowego.

Myślę, że dalej nie muszę przekonywać Czytelników o istocie tego zjawiska. Zagadnienie to leży na drugim biegunie rozwoju mikroinformatyki. Tysiące komputerów osobistych nigdy jednak nie zastąpią wysokosprawnej sieci komputerowej. Natomiast budowa takiej sieci i jej prawidłowa eksploatacja wymaga znacznych nakładów oraz dobrze przygotowanej infrastruktury — niezawodnej publicznej lub dedykowanej sieci telefonicznej i nowoczesnych, szybkich komputerów, a także dużych zespołów fachowców znających te zagadnienia.

Obecnie w Polsce nie istnieje żadna sieć komputerowa z prawdziwego zdarzenia. Lokalnie rozproszonych terminali systemu CYBER lub lepiej wyposażonych komputerów IBM, RIAD czy ODRY 1305 nie można również traktować jako sieci. Brakuje też przy tym prawie wszystkiego, co jest potrzebne do budowy takiej sieci.

W INFORMATYCE możemy jedynie wypełnić lukę informacyjną. Stąd też wykorzystując krajowe pionierskie prace badawcze w tej dziedzinie, prowadzone w Politechnice Wrocławskiej, prezentujemy pierwszy artykuł poświęcony problemom budowy krajowej Międzyuczelnianej Sieci Komputerowej (MSK). Artykuł ten przede wszystkim wprowadza Czytelnika w tematykę sieci komputerowych, przedstawia model standardowej sieci rekomendowanej przez ISO (International Standard Organization). Model ten najpełniej uwzględnia dotychczasowe doświadczenia i przybliża wizję architektury przyszłych sieci komputerowych.

W kolejnych numerach INFORMATYKI będziemy przedstawiać doświadczenia z projektowania i eksploatacji MSK. (WI)

EUGENIUSZ BILSKI
Instytut Cybernetyki Technicznej
Politechnika Wrocławska

ZBIGNIEW FRYŹLEWICZ
ZBIGNIEW HUZAR
ANDRZEJ KALIŚ
Centrum Obliczeniowe
Politechnika Wrocławska

Rozległe sieci komputerowe

Sieci komputerowe mają krótką, zaledwie kilkunastoletnią historię. Początek prac w tym zakresie kojarzy się często z wprowadzeniem koncepcji transmisji danych z komutacją pakietów (1964 rok). Szeroko zakrojone prace zostały podjęte już pod koniec lat sześćdziesiątych, prawie niezależnie od siebie, najpierw w National Physical Laboratory w Wielkiej Brytanii oraz, w krótkim odstępie czasu, w USA w ramach powszechnie znanego projektu sieci ARPA (Advanced Research Projects Agency).

Sieci komputerowe to zespół systemów komputerowych (nazywanych też komputerami obliczeniowymi) połączonych ze sobą wspólną podsiecią komunikacyjną. W trakcie rozwoju sieci komputerowych wykształciły się dwa typy sieci, najpierw sieci rozległe (WAN — Wide Area Networks) a następnie sieci lokalne (LAN — Local Area Networks). Zasadnicze różnice pomiędzy obu typami wynikają w równym stopniu z różnic w budowie i funkcjonowaniu ich podsieci komunikacyjnych.

Podsieci komunikacyjne dla sieci typu WAN, pracujące z reguły w oparciu o zasadę komutacji pakietów, składają się z węzłów (zwykle specjalizowanych minikomputerów lub systemów wielomikroprocesorowych) połączonych ze sobą kanałami telekomunikacyjnymi, wykorzystującymi łącza telefoniczne i radiowe (w tym satelitarne). Dzięki temu można budować sieci obejmujące swym zasięgiem praktycznie cały obszar kuli ziemskiej.

Podsieci komunikacyjne w sieciach typu LAN mają znacznie prostszą budowę. Najczęściej medium transmisyjnym jest kabel koncentryczny (ostatnio coraz częściej światłowód), do którego mają dostęp, za pośrednictwem odpowiednich układów logicznych, stacje robocze, którymi mogą być komputery, pamięci, urządzenia wejścia-wyjścia. Istotną cechą takich rozwiązań jest możliwość prowadzenia szybkiej transmisji (1—100 Mb/s), na ogół niższe koszty budowy, ale też znaczna ograniczoność przestrzenna (odległość pomiędzy najbardziej oddalonymi stacjami nie może przekraczać kilkunastu kilometrów).

W niniejszym artykule rozważania dotyczą wyłącznie sieci rozległych, pominięto analizę różnic i podobieństw sieci WAN i LAN.

Obecnie funkcjonuje na świecie ponad 200 sieci rozległych. Są to sieci o różnym charakterze użytkowym: budowane przez poszczególne kraje w ramach rozwoju usług telekomunikacyjnych (np. TRANSPEC we Francji, 1978, PSS w Wielkiej Brytanii, 1981), przez grupy krajów (np. EURONET, zbudowana przez kraje EWG, 1980; NPSDN,

zbudowana przez kraje skandynawskie, 1980), oraz przez międzynarodowe zrzeszenia (np. SITA — sieć organizacji lotniczych, 1971; SWIFT — sieć organizacji bankowych, 1976) itp. Można też zaobserwować nową tendencję związaną z rozwojem sieci — przechodzenie na wyższy jakościowo poziom. Jest to dążenie — co najmniej w sferze możliwości technicznych i użytkowych — do integracji oddzielnych sieci w makrosieci. Zasadniczą barierą stają się tutaj problemy prawne i ekonomiczne, a nawet polityczne.

Sieci komputerowe są nowym narzędziem informatycznym. Oferują one użytkownikowi znacznie poszerzony zakres usług.

- Następuje uogólnienie tradycyjnych usług wielodostępu. Użytkownik terminala lub stacji abonenckiej, poza dotychczasowym dostępem do pojedynczego systemu komputerowego, uzyskuje dodatkowo możliwość dostępu do dowolnie wybranego komputera obliczeniowego sieci.

- Oferowana jest nowa kategoria usług w postaci przetwarzania rozproszonego. Aby wykonać zadanie użytkownik może korzystać z wielu komputerów obliczeniowych. W szczególności realizacja poszczególnych fragmentów zdekomponowanego zadania może przebiegać równolegle w różnych komputerach.

- Dzięki możliwości przetwarzania rozproszonego wykształciły się co najmniej dwa dalsze typy usług. Są to dystrybucja i wykorzystanie rozproszonych baz danych oraz poczta elektroniczna.

Najbardziej jednak istotną, na początku nieoczekiwaną konsekwencją wprowadzania sieci komputerowych jest nie tylko rozszerzenie, ale przede wszystkim podniesienie poziomu jakości usług informatycznych. Stało się tak dlatego, że tylko dzięki sieciom mogły nastąpić podziały i głęboka specjalizacja zespołów tworzących oprogramowanie oferowane w sieci. Zaskakujące jest również i to, że właśnie utrzymanie takich zespołów, rozwijających i pielęgnujących oprogramowanie sieci, stanowi obecnie najdroższy składnik kosztów budowy i eksploatacji sieci.

MODEL SIECI KOMPUTEROWEJ ISO/OSI

Większość firm komputerowych oraz prawie wszystkie organizacje związane z zagadnieniami przesyłania i przetwarzania informacji utworzyły swoje modele sieci komputerowych. Modelem, który zyskał największy rozgłos i uznanie wśród profesionalistów jest model przedstawiony przez ISO, nazywany Modelem Odniesienia Współdziałania Systemów Otwartych (Reference Model of Open Systems Interconnection, w skrócie OSI).

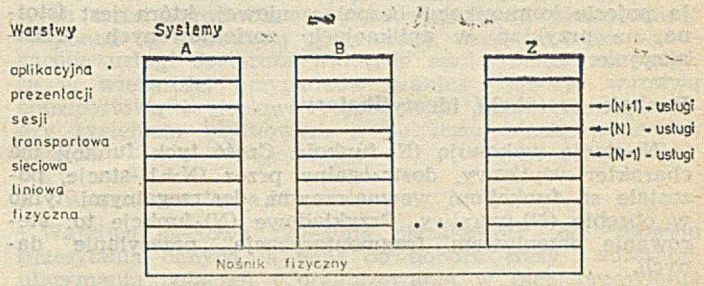
Głównym celem tego modelu było dokonanie standaryzacji zasad współdziałania wzajemnie połączonych systemów komputerowych — zwykle o różnej architekturze. Standaryzacji podlega tylko zewnętrznie obserwowalne zachowanie tych systemów. Stąd ich nazwa **otwarte**, czyli takie, które swobodnie komunikują się z innymi, w przeciwieństwie do systemów zamkniętych, poprawnie funkcjonujących w swoim indywidualnym środowisku. Wewnętrzna organizacja i funkcjonowanie każdego systemu, który ma atrybut otwartości, są nieistotne, ponieważ cechy te nie są zauważalne przez innych.

Model ISO/OSI ma obecnie status standardu miedzynarodowego (IS — International Standard) i jest opisany w dokumencie ISO IS 7498.

Systemy, warstwy, stacje

Model odniesienia ISO jest abstrakcyjnym opisem komunikacji międzyprocesorowej. Opisuje on procesy aplikacyjne realizowane w różnych, odległych od siebie systemach. Za system uważa się tutaj autonomiczny komputer(y) wraz z oprogramowaniem i urządzeniami wejścia-wyjścia oraz użytkowników, którzy przetwarzają lub przesyłają informację.

Każdy indywidualny system jest widziany jako logiczne połączenie uporządkowanego hierarchicznie zbioru podsystemów. Sąsiednie podsystemy współdziałają dzięki łączącym je interfejsom. Podsystemy tej samej rangi tworzą warstwę. Dla uproszczenia nazewnictwa określa się ją mianem (N)-warstwy, przy czym warstwę wyższą nazywa się (N+1)-warstwą, a niższą (N—1)-warstwą (rys. 1).



Rys. 1. Systemy, warstwy, usługi

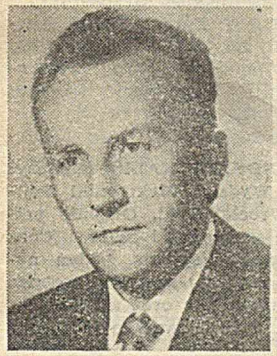
W (N)-warstwie działają (N)-stacje, które reprezentują zdolności przetwarzania rozproszonego tej warstwy w realizacji jej funkcji.

W modelu ISO/OSI wyróżnia się siedem warstw. Są to warstwy — poczynając od najniższej — fizyczna, liniowa, sieciowa, transportowa, sesji, prezentacji i aplikacji.

Usługi i punkty dostępu do usług

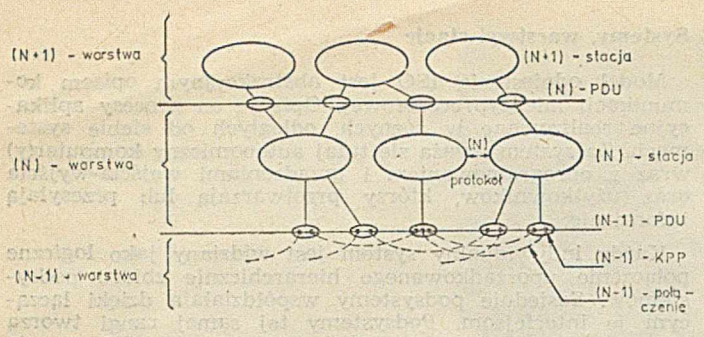
(N)-stacje rozproszone w (N)-warstwie, pośród współdziałających systemów otwartych, wspólnie realizują (N)--usługę dla (N+1)-stacji (wyjątkiem jest warstwa najwyższa, która jest tylko biorcą usług). Inaczej ujmując, (N)--stacje dodają nową usługę do (N—1)-usługi realizowanej przez (N—1)-warstwę i tak wzbogaconą (N)-usługę oferują (N+1)-stacjom. (N)-usługi są oferowane (N+1)-stacjom poprzez (N)-punkty dostępu do usług, w skrócie PDU (Service Access Points). (N)-PDU reprezentują logiczne interfejsy pomiędzy (N)-stacjami i (N+1)-stacjami — rys. 2.

Podstawową usługą oferowaną przez (N)-warstwę (N+1)--warstwie jest realizacja (N)-połączenia (asocjacji) pomiędzy (N)-PDU. (N)-połączenie realizowane przez (N)-warstwę, na życzenie (N+1)-warstwy, umożliwia (N+1)-stacjom nawiązanie i prowadzenie diologu, podczas którego stacje te mogą wymieniać interesujące je dane. W modelu są dopuszczalne połączenia dwu- i wielopunktowe. Zakończenie (N)-połączenia w (N)-PDU jest nazywane końcowym punktem (N)-połączenia, w skrócie (N)-KPP, ((N)-Connection End Point). Pomiędzy tą samą parą (N)-PDU może jednocześnie współistnieć wiele połączeń i każde z nich jest na swoim końcu wyróżniane identyfikatorem rozpoznawanym przez (N)-stację i (N+1)-stację.



Mgr inż. EUGENIUSZ BILSKI ukończył w 1957 r. studia na Wydziale Łączności Politechniki Wrocławskiej. W latach 1959—1971 pracował w WZE ELWRO, ostatnio jako zastępca dyr. ds. technicznych. Od 1971 r. pracuje w Politechnice Wrocławskiej, jako główny projektant systemu WASC, a obecnie Międzyuczelnianej Sieci Komputerowej MSK.

Dr inż. ZBIGNIEW FRYŻLEWICZ ukończył w 1972 r. Wydział Elektroniki Politechniki Wrocławskiej. W latach 1979—1983 był czynnie zaangażowany w projektowaniu i implementacji sieci komputerowej MSK. Pracuje w Centrum Obliczeniowym PWr.



Rys. 2. Stacje, punkty dostępu do usług, końcowe punkty połączeń, protokoły

Najnowsze edycje opisujące model ISO/OSI wprowadzają pojęcie komunikacji bezpołączeniowej, która jest istotna, na przykład, w aplikacjach zorientowanych transakcyjnie.

Funkcje, protokoły, identyfikatory

(N)-stacje wykonują (N)-funkcje. Część tych funkcji ma charakter użytkowy, dostrzegalny przez (N+1)-stacje, pozostałe są funkcjami wewnętrznymi, dostrzegalnymi tylko w obrębie (N)-warstwy. Przykładowe (N)-funkcje to: sterowanie przepływem, fragmentaryzacja, przesyłanie danych.

Współpraca pomiędzy (N)-stacjami jest realizowana poprzez (N)-protokół(y).

(N)-protokół jest zbiorem formatów i reguł regulującym komunikację pomiędzy (N)-stacjami realizującymi (N)-funkcje. (N)-funkcje dodając się do (N—1)-usług tworzą (N)-usługi oferowane (N+1)-stacjom.

Komunikacja pomiędzy (N)-stacjami jest realizowana wyłącznie za pomocą (N—1)-usługi. Komunikacja bezpośrednia pomiędzy, na przykład, (N)-stacjami ulokowanymi w tym samym systemie dla dostępu do lokalnie dzielonych zasobów, nie jest widoczna z zewnątrz tego systemu i tym samym nie jest rozważana w architekturze OSI.

Stacje najniższej warstwy komunikują się poprzez nośnik (medium) fizyczny, uważany jako tworzący warstwę 0.

Jak już nadmieniono, (N)-stacja i (N+1)-stacja wyróżnia pojedyncze (N)-połączenia poprzez identyfikator związany z (N)-KPP. W modelu ISO/OSI definiuje się zarówno identyfikatory dla obiektów wewnątrz warstw, jak i leżących na granicy pomiędzy sąsiednimi warstwami, a także podaje się relacje jakie istnieją między tymi identyfikatorami.

Operacje na połączeniach

A. Zakładanie i zwalnianie połączeń. (N+1)-stacja żądając zestawienia (N)-połączenia przekazuje do lokalnego (N)-PDU (N)-adres odległego (N)-PDU, z którym to połączenie ma być ustanowione. Gdy zadane (N)-połączenie zostaje zestawione, to zarówno (N+1)-stacja jak i (N)-stacja używają swoich (N)-KPP identyfikatorów dla lokalnego wyróżniania tego połączenia. (N)-połączenia mogą być zakładane i zwalniane dynamicznie, na bazie już istniejących (N—1)-połączeń. Zwykle założenie (N)-połączenia musi być poprzedzone istnieniem (N—1)-połączenia. Założenie (N—1)-połączenia wymaga istnienia (N—2)-połączenia itd.

W pewnych przypadkach (N)-połączenie może być zakładane jednocześnie z zakładaniem (N—1)-połączenia. Sytuacja taka jest dopuszczalna wtedy, gdy usługa ustawiania (N—1)-połączeń pozwala (N)-stacjom na wymianę informacji koniecznych dla zestawienia (N)-połączeń.

B. Multipleksowanie połączeń. Wyróżnia się trzy typy konstrukcji (N)-połączeń na bazie (N—1)-połączeń. Są one następujące (rys. 3):

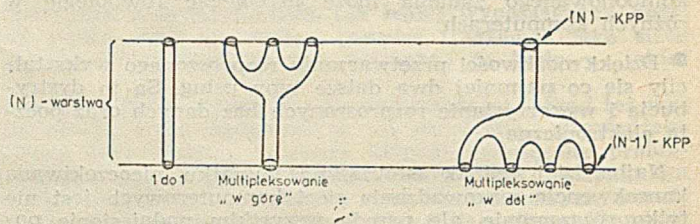
a) pojedyncze (N)-połączenia, zbudowane na pojedynczym (N—1)-połączeniu

b) kilka (N)-połączeń zbudowanych na wspólnym, pojedynczym (N—1)-połączeniu

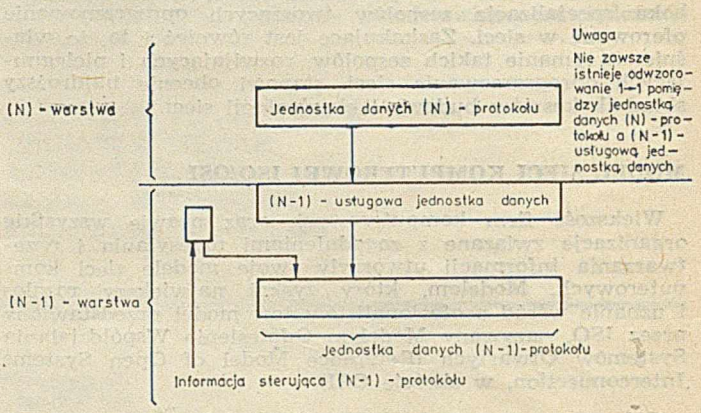
c) pojedyncze (N)-połączenie zbudowane na kilku (N—1)-połączeniach.

Podane schematy wzajemnych powiązań (N) i (N—1)-połączeń mogą być używane przy:

- optymalizacji kosztów przesyłania
- zwiększaniu przepustowości
- zwiększaniu niezawodności.



Rys. 3. Zależności między (N) i (N-1)-połączeniami



Rys. 4. Relacje logiczne pomiędzy jednostkami danych w przyległych warstwach

C. Przesyłanie danych w połączeniu. Informacja, zarówno pomiędzy stacjami w warstwie, jak i pomiędzy stacjami związanymi wspólnym PDU, jest przesyłana w różnych typach jednostek danych. Powiązania pomiędzy niektórymi jednostkami przedstawiono w tablicy 1 oraz na rys. 4. Definicje wprowadzanych tam pojęć są następujące:

- **informacja sterująca (N)-protokołu** jest to informacja wymieniana pomiędzy dwoma (N)-stacjami, z wykorzystaniem (N—1)-połączeń, dla koordynacji ich współpracy

Dr inż. ZBIGNIEW HUZAR ukończył studia na Wydziale Elektroniki Politechniki Wrocławskiej. W latach 1969—1978 pracował w Instytucie Cybernetyki Technicznej, a następnie w Centrum Obliczeniowym Politechniki Wrocławskiej. Obecnie pełni funkcję dyrektora Centrum Obliczeniowego. Zajmuje się zagadnieniami programowania współbieżnego oraz oprogramowania sieci komputerowych.

Dr inż. ANDRZEJ KALIŚ ukończył w 1975 r. Wydział Elektroniki Politechniki Wrocławskiej. Obecnie pełni funkcję zastępcy dyrektora Centrum Obliczeniowego ds. badań naukowych. Zajmuje się diagnostyką układów progowych oraz niekonwencjonalnymi strukturami procesorów.

● **(N)-dane użytkowe** są to dane przesyłane pomiędzy dwoma (N)-stacjami w imieniu (N+1)-stacji, dla których to stacji (N)-stacje zapewniają usługę

● **jednostki danych (N)-protokołu** są to jednostki danych, które zawierają informację sterującą (N)-protokołu i (zwykle) (N)-dane użytkowe

● **informacja sterująca (N)-interfejsu** jest to informacja wymieniona pomiędzy stacjami (N+1) i (N) dla koordynowania wzajemnej współpracy

● **jednostka danych (N)-interfejsu** jest jednostką informacji przesyłaną przez punkt dostępu do usług pomiędzy stacjami (N+1) i (N) w pojedynczej interakcji

● **(N)-usługowa jednostka danych** jest pewną wielkością danych (N)-interfejsu, która zachowuje integralność przy przesyłaniu ich z jednego końca (N)-połączenia w drugi; dane mogą być przetrzymywane w połączeniu aż do momentu przesyłania do tego połączenia kompletnej usługowej jednostki danych

● **(N)-usługowa jednostka danych przyspieszonych** jest małą (N)-usługową jednostką danych, której przesłanie jest przyspieszone w stosunku do normalnego strumienia danych; (N)-warstwa, przy przesyłaniu tych danych, zapewnia ich odpowiednie umiejscowienie w stosunku do całości strumienia danych. Dane przyspieszone służą zwykle rozwiązywaniu problemów sterowania

Tabela 1. Powiązania pomiędzy jednostkami danych

	Sterowanie	Dane	Łącznie
(N)-(N) stacje w warstwie	informacja sterująca (N)-protokołu	(N)-dane użytkowe	jednostka danych (N)-protokołu
(N)-(N-1) przyległe warstwy	informacja sterująca (N-1)-interfejsu	dane — (N-1)-interfejsu	jednostka danych (N-1)-interfejsu

Siedmiowarstwowy model ISO/OSI

Opisane podstawowe pojęcia i elementy abstrakcyjne modelu służą jako moduły umożliwiające skonstruowanie modelu komunikacji międzyprocesowej. Zagadnienie tej komunikacji jest przedstawione w formie współpracy siedmiu warstw o szczegółowo określonych funkcjach i hierarchicznym uporządkowaniu.

A. Warstwa aplikacji

Warstwa aplikacji jest warstwą najwyższą. Reprezentuje ona właściwe przetwarzanie informacji — procesy aplikacyjne.

Procesy aplikacyjne są praprzyczyną sprawczą wymiany informacji we wszystkich pozostałych warstwach, są one ostatecznym źródłem i odbiorcą wymienianych danych. Procesy aplikacyjne mogą reprezentować zadania użytkowników sieci komputerowej, a także reprezentować czynności związane ze sterowaniem i zarządzaniem pracą sieci.

Procesy aplikacyjne komunikują się ze sobą i wykorzystują w tym celu protokoły warstwy aplikacyjnej. Te aspekty procesów aplikacyjnych, które są związane z komunikacją międzyprocesową, tworzą stację aplikacyjną będącą elementem modelu ISO/OSI. Pozostała część procesu aplikacyjnego jest poza zainteresowaniem tego modelu.

B. Warstwa prezentacji

Głównym zadaniem warstwy prezentacji jest zapewnienie niezależności procesów aplikacyjnych od różnic w reprezentacji danych (od leksyki i syntaktyki). Protokół zwarty prezentacji umożliwia użytkownikowi wybór kontekstu prezentacji. Użytkownicy OSI mogą używać istniejących i proponowanych przez ISO kontekstów lub też mają możliwość ustanowienia swoich własnych i ewentualnego zarejestrowania ich w ISO.

C. Warstwa sesji

Głównym celem funkcjonowania warstwy sesji jest zapewnienie mechanizmów dla organizacji i strukturalizacji interakcji pomiędzy procesami aplikacyjnymi.

Mechanizmy warstwy sesji umożliwiają procesom aplikacyjnym na zestawienie lub rozłączenie połączeń oraz na transfer danych w ustalonym trybie (dupleksowym lub półdupleksowym) z ustalonym sposobem synchronizacji i strukturalizacji wymienianych danych.

D. Warstwa transportowa

Zasadniczym celem funkcjonowania warstwy transportowej jest przesłanianie procesom aplikacyjnym sposobu działania podsieci komunikacyjnej. Połączenia transportowe dostarczane przez tę warstwę są widziane przez warstwy wyższe tylko jako pewne kanały łączności o określonej przepustowości, opóźnienia i niezawodności.

Warstwa transportowa ma za zadanie transparentne przesyłanie danych z systemu-źródła do systemu-adresata. Uwalnia ona wyższe warstwy od organizowania niezawodnego i efektywnego, z punktu widzenia kosztów, przesyłania danych.

Warstwa transportowa optymalizuje użycie usług sieciowych oraz wprowadza (zwykle) dodatkową niezawodność — ponad tę, oferowaną przez usługi sieciowe) i tak wzbogacone usługi dostarcza warstwie sesji. Należy zauważyć, że w większości przypadków granica między warstwą transportową i sieciową reprezentuje także granicę między systemem użytkowym a publiczną siecią transmisji danych.

E. Warstwa sieciowa

Warstwa sieciowa zapewnia niezależność od sposobu przesyłania danych, a więc od doboru trasy, adresacji, utrzymania połączeń i organizowania w nich przesyłania danych. Wszystkie te funkcje są w warstwie sieciowej realizowane, ale maskowane dla warstwy transportowej. W ten sposób warstwa transportowa otrzymuje tylko informacje o jakości usług i ich koszcie, nie „widząc" czy przesyła się dane po łączach światłowodowych czy telefonicznych, czy używa się komutacji pakietów czy też innej techniki multipleksowania.

F. Warstwa liniowa

Zadaniem warstwy liniowej jest zapewnienie funkcjonalnych i proceduralnych środków do przesyłania danych pomiędzy stacjami sieciowymi, a także do wykrywania i usuwania błędów, jakie mogą wystąpić w warstwie fizycznej. Funkcje tej warstwy są silnie uzależnione od technologii fizycznego przesyłania danych.

Zagwarantowanie wysokiej efektywności przesyłania danych wymaga istnienia różnych protokołów, odpowiednich do stosowanej technologii i typów połączeń.

G. Warstwa fizyczna

Zapewnia ona mechaniczne, elektryczne, funkcjonalne i proceduralne dopasowanie do fizycznego medium transmisyjnego.

PROTOKOŁY DLA MODELU ISO/OSI

Model architektury ISO/OSI, definiując usługi każdej warstwy i sposób udostępnienia tych usług warstwie wyższej, proponuje też ogólną koncepcję specyfikowania funkcji wypełnianych przez każdą warstwę. Szczegółowe funkcjonowanie każdej warstwy jest zdefiniowane przez specyficzne dla niej protokoły. Poniższy przegląd zawiera krótką charakterystykę zbioru protokołów rozważanych przez ISO. Kształt tych protokołów nie zawsze jest do końca określony. Dotyczy to w szczególności dwóch najwyższych warstw, gdzie jeszcze nie zamknięto dyskusji na ich temat. Protokoły niższych warstw, chociaż już w pewien sposób ustalone, będą podlegały zmianom i modyfikacjom zgodnym z postępem w technologii.

Protokoły w warstwie fizycznej

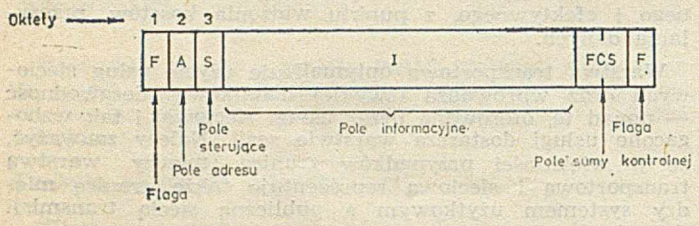
Adaptowano tutaj standardy wypracowane przez CCITT. Oznacza się je jako X21 bis i X21. Pierwszy z nich to nic innego jak powszechnie znany V.24 i nierozłącznie związane z nim inne standardy, które dopiero w całości definiują mechaniczne, elektryczne, funkcjonalne i proceduralne cechy interfejsu z fizycznym medium transmisyjnym. Standard drugi (X.21) został w CCITT opracowany na początku lat siedemdziesiątych, a więc wtedy, gdy model ISO/OSI jeszcze nie istniał. Standard ten opisuje technikę transmisji zwaną komutacją obwodów (ang. circuit switching). Jeżeli obecnie mówi się o użyciu tego standardu jako obowiązującego na poziomie fizycznym, to ma się na myśli tylko i wyłącznie jego fragment definiujący

podane wyżej cechy interfejsu. Wymaga to odwołania i do innych standardów serii X, jak np. X.24, zawierającego opis funkcjonalny obwodów interfejsu.

Protokoły w warstwie liniowej

W obecnej chwili dominującą rolę zyskały protokoły zorientowane bitowo, a szczególnie preferowany przez ISO i CCITT protokół LAPB, nazwany także X.25.2. W swojej istocie jest to klasa protokołu wyodrębniona z ogólnej procedury (protokołu) HDLC zdefiniowanej przez ISO.

Używając pojęć modelu, jednostką danych protokołu jest tutaj ramka o ogólnej strukturze jak na rys. 5.



Rys. 5. Struktura ramki w protokole LAPB

Wymieniona klasa, tworząca protokół LAPB, oparta jest na następujących założeniach:

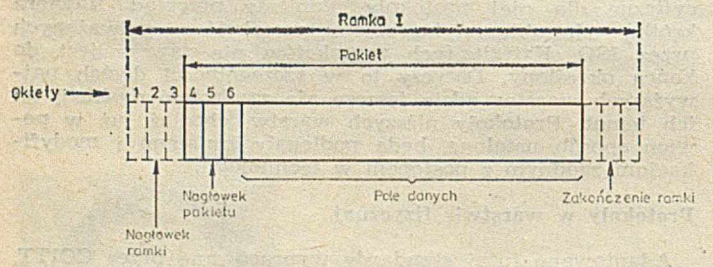
- współpracujące stacje działają w zrównoważonym, asynchronicznym trybie odpowiedzi
- stacje używają bazowego repertuaru rozkazów lub odpowiedzi z rozszerzeniem o ramkę odrzucenia i żądania retransmisji (REJECT)
- ramki informacyjne (ramki I) używane są tylko jako rozkazy.

Główną funkcją, którą ten protokół realizuje jest pozbawione błędów przesyłanie bloków informacji (ramek I) pomiędzy sąsiednimi systemami. Oprócz tego protokół realizuje sterowanie przepływem danych poprzez wbudowany mechanizm okna. Nowe technologie są przyczyną powstawania nowych protokołów liniowych. W szczególności można oczekiwać finalizowania rozwoju standardów dla transmisji równoległej i standardów dla transmisji szeregowej prowadzonej ze względu na niezawodność lub przepustowość w kilku liniach równolegle.

Protokoły w warstwie sieciowej

Szeroko akceptowaną propozycją protokołu w warstwie sieciowej jest poziom 3 protokołu X.25, nazywany też protokołem pakietowym lub X.25.3. Standard tego protokołu został opracowany przez CCITT także przed rozwojem modelu ISO/OSI i w swojej istocie opisuje on interfejs pomiędzy użytkownikiem a podsiecią komutacji pakietów, nie definiując jawnie protokołu istniejącego pomiędzy kolejnymi dwoma wewnętrznymi węzłami tej podsieci.

Używając pojęć modelu, jednostką danych protokołu jest tu pakiet (rys. 6) lub wprowadzony w 1981 r. datagram.



Rys. 6. Struktura pakietu informacyjnego na tle ramki protokołu LAPB

Pakiety są używane do zestawiania, zerowania lub zamykania połączeń sieciowych stałych lub czasowych i przesyłania w nich danych. W nomenklaturze protokołu X.25.3. połączenia te są nazywane wirtualnymi. Główna funkcja protokołu to przesyłanie pakietów lub datagramów pomiędzy sąsiednimi węzłami w stronę odbiorcy. Wymaga ona między innymi realizacji funkcji marszrutyzacji i aktywnego sterowania przepływem, dla przeciwdziałania potencjalnym przeciążeniom węzłów. Realizacja tej ostatniej funkcji następuje przez stosowanie odpowiednich pakietów i wbudowany w protokół mechanizm okna. Połączenia wirtualne budowane za pomocą pakietów odpowiadają w modelu ISO/OSI transmisji połączeniowej. Drugi typ transmisji — bezpołączeniowej, można realizować stosując datagramy.

Protokół w warstwie transportowej

Szeroko akceptowaną propozycją w tej warstwie jest protokół transportowy, opracowany przez ISO DIS 8077, uwzględniający także prace prowadzone w ramach CCITT i ECMA.

Wyróżnia się w nim pięć różnych klas połączeń transportowych o różnym stopniu wymagań co do niezawodności, przepustowości i kosztów. Połączenia te są budowane na połączeniach sieciowych dowolnej klasy.

Połączenia transportowe w klasach 0 i 1 są realizowane na pojedynczym połączeniu sieciowym, a w klasach 2, 3, i 4 mogą być multipleksowane. Dodatkowo, klasy 1, 3, 4 wprowadzają niezależność połączeń transportowych od sieciowych. Bez utraty połączeń transportowych może nastąpić przełączenie połączeń sieciowych, gdy aktualnie używane nie funkcjonują lub funkcjonują źle. Klasa 4 z wybranymi opcjami wprowadza dodatkowe zabezpieczenia przed błędami.

Tak zdefiniowany protokół daje możliwości realizacji wszystkich funkcji związanych ze sterowaniem przesyłania danych niezawodnym i optymalnym, jeśli chodzi o koszty.

Używając pojęć modelu, jednostkami danych protokołu są tutaj komunikaty danych lub komunikaty sterujące, które — w zależności od używanej klasy — są w różny sposób przenoszone przez jednostki danych protokołu sieciowego.

Protokół w warstwie sesji

Obecna wersja protokołu, DP 8327, opracowana przez ISO dla tej warstwy, wyrosła z wcześniejszych propozycji ECMA i równoległych prac w CCITT, w szczególności prac nad systemem TELETEX. Z całego protokołu opisanego w tym dokumencie można implementować jeden z trzech funkcjonalnie pełnych podzbiorów oznaczonych kolejno BCS (Basic Combined Subset), BSS (Basic Synchronized Subset) i BAS (Basic Activity Subset).

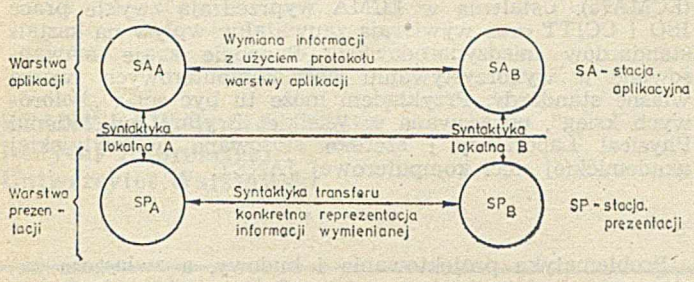
Podzbiór BCS realizuje podstawowe funkcje warstwy sesji, zarówno dla modelu transmisji dupleksowej, jak i półdupleksowej. Jest on przeznaczony dla zastosowań, w których nie jest wymagana synchronizacja.

Podzbiór BSS zawiera wszystkie funkcje BCS, a dodatkowe jego możliwości to synchronizacja według małych i dużych punktów synchronizacyjnych, resynchronizacja, negocjacja przekazywania upoważnień i przesyłanie ograniczonej wielkości danych, niezależnie od posiadania upoważnienia. Podzbiór ten jest przeznaczony do obsługi złożonych aplikacji, w których wymaga się synchronizacji w przesyłanych strumieniach danych. Podzbiór BAS zawiera wszystkie możliwości BSS oraz dodatkowo umożliwia użytkownikom sesji przekazywanie informacji o stanach wyjątkowych (ang. exeption reporting), nie przenoszonych przez protokoły i sterowania efektywnością, czyli rozbicia całego dialogu na podstrumienie (aktywności) i odrębnego sterowania każdym z nich (ang. activity reporting). Ten podzbiór protokołu z odpowiednio dobranymi parametrami zapewnia obsługę zastosowań proponowanych przez CCITT.

Protokół w warstwie prezentacji

W początkowej fazie rozwoju protokołów dla modelu ISO/OSI zakładano, że istniejące zasadnicze różnice w strumieniach informacji generowanych przez takie procesy, jak wirtualny terminal transmisji zbiorów lub zadań, wymagają odrębnych protokołów, zarówno w warstwie prezentacji, jak i aplikacji. Obecny (z przełomu lat 1983—1984) model usług prezentacji i aplikacji jest inny. Zakłada on istnienie jednolitego protokołu w warstwie prezentacji oraz istnienie wspólnych procedur wykorzystywanych przez protokoły aplikacyjne. Model powiązań między warstwami prezentacji i aplikacji przedstawia rys. 7.

Granica podziału między warstwami przebiega w ten sposób, że w warstwie prezentacji są rozstrzygane wszystkie kwestie związane z uzgodnieniem i poprawnym rozpoznaniem syntaktyki informacji, przesyłanej między stacjami odległych systemów, natomiast w warstwie aplikacji rozstrzygane są kwestie semantyczne.



Rys. 7. Powiązania między warstwami aplikacji i prezentacji

Warstwa prezentacji realizuje żądania usług płynące z warstwy aplikacji bądź udostępnia tej warstwie usługi z warstwy sesji. Usługi te są dzielone na siedem kategorii nazywanych udogodnieniami. Udogodnienia dzielą się na elementy usługowe, te zaś — na operacje elementarne. Operacje elementarne są odwzorowywane w działania protokołu.

Działania te (w ramach wymienionych siedmiu udogodnień) są następujące:

Udogodnienie **Połączenie** realizuje połączenie wybranych punktów dostępu do usług, pomiędzy dwiema stacjami aplikacyjnymi. Udogodnienie **Kontekst** umożliwia warstwie prezentacji dopasowanie jej środowiska (kontekstu) dla wymienianej informacji użytkowej. Kontekst ten składa się zarówno ze składni transferu jak i stanu warstwy prezentacji. Udogodnienie **Transmisja danych** jest środkiem, za pomocą którego warstwa aplikacji przesyła swoje dane pomiędzy współdziałającymi stacjami. Udogodnienie **Zarządzanie dialogiem i synchronizacją** udostępniają warstwie aplikacji **Widzenie** odpowiednich usług warstwy sesji. Udogodnienie **Przerwanie** zapewnia możliwość zawieszania, a następnie ponownego podjęcia komunikacji między systemami. Tym sposobem rzeczywisty system i zasoby komunikacyjne mogą być chwilowo udostępniane na inne potrzeby. Udogodnienie **Rozłącz** zapewnia środki do rozłączania połączenia zestawionego z użyciem udogodnienia **Połączenie.**

Opis modelu usług warstwy prezentacji i języka (protokołu) prezentacji jest zawarty w opracowaniach ISO/ /TC97/SC16 nr 1828 i 1829, które mają obecnie status projektów norm (DP — Draft Proposal).

Struktura stacji aplikacyjnej i protokoły aplikacyjne

Warstwa aplikacyjna jest warstwą najwyższą. W warstwie tej istnieją identyfikowalne elementy usługowe, które przyjmują od procesów użytkowych żądania realizacji zaleceń w środowisku OSI i przekazują odpowiedzi jako rezultaty żądanych działań. Podzbiory tych elementów usługowych tworzą unikalne typy stacji aplikacyjnych. Przykładem trzech zbiorów elementów usługowych, z których każdy tworzy odrębny typ stacji aplikacyjnej, są: Usługa transferu zbiorów, Usługa wirtualnego terminala, Usługa transferu zadań. W strukturze każdej stacji rozpoznaje się istnienie trzech kategorii elementów usługowych: CASE, SASE i UE.

Wspólne Aplikacyjne Elementy Usługowe (CASE — Common Application Service Elements) zapewniają realizację usług niezależnych od natury aplikacji, jak na przykład zestawianie lub rozłączanie współdziałania pomiędzy procesami aplikacyjnymi. Specyficzne Aplikacyjne Elementy Usługowe (SASE — Specific Application Service Elements) realizują żądania przesyłania informacji (np. przesyłanie zbioru, dostęp do bazy, transfer zadania) lub realizują usługi dla specyficznych procesów aplikacyjnych (np. bankowych, zapytaniowych). Każda stacja aplikacyjna zawiera jeszcze Elementy Usługowe (UE — User Element) reprezentujące interfejs pozostałej części procesu aplikacyjnego do elementów usługowych warstwy aplikacji.

W chwili obecnej jest wstępnie ustalony zbiór usług tworzących CASE oraz zaawansowane są prace nad usługami SASE i związanymi z nimi takimi protokołami, jak: Transfer Zbiorów, Wirtualny Terminal, Transfer Zadań.

UZUPEŁNIENIA DO MODELU ISO/OSI

Przedstawiony model odniesienia ISO/OSI dotyczy głównie problemów standaryzacji zasad współdziałania systemów otwartych i pozostawia wiele problemów nie rozstrzygniętych. Tymczasem dotychczasowa praktyka projektowania i budowy sieci komputerowych narzucała konieczność przyjmowania różnych ustaleń. Poniżej zostanie zarysowane pole nieokreśloności modelu oraz pokazane pewne tendencje wyrosłe z praktyki.

Punktem wyjścia jest stwierdzenie, że sieć komputerowa ma świadczyć swym użytkownikom pewne usługi obliczeniowe. Umownie, z punktu widzenia użytkownika, można wyróżnić następujące trzy poziomy usług: przetwarzania skupionego, rozproszonego i wirtualnego.

Na poziomie usług przetwarzania skupionego użytkownik dysponuje możliwością nawiązania połączenia z dowolnym komputerem obliczeniowym sieci i podjęcia z nim współpracy tak, jakby był jego lokalnym użytkownikiem. Dodatkowo użytkownik może polecać realizację przesłania swych zbiorów pomiędzy komputerem, z którym współpracuje, a wskazanym przez niego innym komputerem obliczeniowym.

Poziom przetwarzania rozproszonego wprowadza pojęcie zadania sieciowego, które można rozumieć jako pewien uporządkowany częściowo zbiór zadań elementarnych. Częściowy porządek, zadany jawnie lub niejawnie, wyznacza dopuszczalną kolejność zadań elementarnych. Zadania elementarne to takie, które mogą być wykonywane w pojedynczym komputerze obliczeniowym. Dysponując możliwością formułowania zadania sieciowego, użytkownik dekomponuje swój problem na zadania elementarne, określa miejsce ich wykonania i sposób współpracy tak, aby uzyskać oczekiwane rozwiązanie.

Poziom przetwarzania wirtualnego maskuje przed użytkownikiem strukturę wewnętrzną sieci. Sieć jawi się dla użytkownika jako jeden, być może bardzo duży, system komputerowy. Korzystając z tego użytkownik definiuje swoje zadanie dla sieci, abstrahując od jej struktury oraz liczby i typów komputerów. Komputer obliczeniowy, który odbiera zadanie od użytkownika, przeprowadza analizę zadania i dekompozycję na zadania sieciowe oraz nadzoruje przebieg ich realizacji. Obecne prace w ISO dotyczą dwóch pierwszych poziomów, natomiast poziom trzeci nie został jeszcze zainicjowany.

Każdy z poziomów usług powinien dysponować odpowiednim językiem do komunikowania się użytkownika z siecią. Języki takie nie są obiektem mieszczącym się w modelu ISO/OSI, tym niemniej w ramach ISO prowadzone są rozważania nad możliwością ich standaryzacji.

Wybrany poziom usług obliczeniowych, a także przyjęty tryb użytkowania sieci, określają funkcje, jakie ma wykonywać komputer obliczeniowy. Funkcje te wygodnie jest podzielić na dwie grupy: funkcje usługowe i sterujące. Funkcje usługowe to te, które realizują bezpośrednio dostrzegane przez użytkownika usługi obliczeniowe, zaś funkcje sterujące to te, które wiążą się z zarządzaniem pracą środowiska wykonawczego dla funkcji usługowych.

Funkcje usługowe

Interesującą tendencją, jaka uwidoczniła się w trakcie rozwoju sieci komputerowych, jest koncepcja grupowania usług wyższych warstw (powyżej warstwy transportowej) modelu ISO/OSI. Do takich usług zalicza się przede wszystkim transfer zbiorów, transfer i manipulację zadań, pocztę elektroniczną i wirtualny terminal. Konsekwencją grupowania usług jest tworzenie protokołów, które — poza elementami komunikacji w warstwie aplikacyjnej — zawierają także elementy prezentacji i sesji. Tego rodzaju koncepcja znalazła — pomijając ostatnio pojawiające się doniesienia o podejmowaniu budowy sieci opartych na „czystej" architekturze ISO/OSI — niemal wyłącznie zastosowanie w dotychczas budowanych lub funkcjonujących sieciach. Wywarła ona też wpływ na prace ISO, gdyż równolegle obok prac nad poszczególnymi warstwami prowadzi się prace dotyczące wymienionych usług. Można spodziewać się, że w dalszym rozwoju tych prac w ramach ISO nastąpi wydzielenie elementów warstwy sesji i prezentacji w usługach zgrupowanych i modyfikacja ich do takich postaci, które wynikają z prac prowadzonych oddzielnie nad każdą z tych warstw.

Funkcje sterujące

Problem sterowania pracą sieci komputerowej można określić abstrakcyjnie jako problem podejmowania decyzji polegających na harmonogramowaniu (szeregowaniu) zleceń użytkowników i takim przydzielaniu do ich realizacji wymaganych zasobów, aby — nie naruszając pewnych ograniczeń — maksymalizować przyjęte kryterium sterowania. Istotnym elementem podejmowania takich decyzji w sieci komputerowej przez każdy z komputerów obliczeniowych, do którego trafia strumień zleceń użytkowników, jest to, że informacja, jaką w określonym momencie dysponuje komputer obliczeniowy o stanie innych komputerów, jest niepełna lub niecałkowicie aktualna. Wynika stąd potrzeba prowadzenia komunikacji pomiędzy komputerami obliczeniowymi w celu uzyskania bieżącej informacji o ich stanie. Utrudnia to znacznie podejmowanie decyzji, a nawet należy przewidywać sytuacje, gdy podjęta decyzja nie może być zrealizowana z powodu różnych, dynamicznie zachodzących zmian w sieci komputerowej.

Zarządzanie pracą sieci komputerowej stanowi obecnie przedmiot intensywnych prac w ISO. Dotychczasowy ich przebieg ujawnia jednak złożoność tych zagadnień i można spodziewać się, że dopiero pod koniec lat osiemdziesiątych pojawią się materiały standaryzacyjne dotyczące — jak wydaje się — dwóch elementów: bazy danych zarządzających, posiadanych przez każdy z systemów sieci, oraz protokołu wymiany danych zarządzających pomiędzy systemami w sieci.

PRACE STANDARYZACYJNE

Standaryzacją w zakresie sieci komputerowych w skali międzynarodowej zajmują się głównie dwie organizacje: ISO i CCITT. Działalność obu organizacji w zasadzie uzupełnia się. ISO przyjęło stanowisko pełnej akceptacji standardów CCITT, opracowanych dla pierwszych trzech warstw modelu ISO/OSI, skupiając się na problemach warstw wyższych. CCITT — poza zainteresowaniem warstwami dolnymi — zajmuje się także problemami warstw wyższych w zakresie usług telematycznych, które są różnymi formami usług pocztowych wykorzystujących komputerowe sieci transmisji danych.

Zagadnieniami standaryzacji sieci komputerowych w ISO zajmuje się specjalnie wyodrębniony podkomitet (TC97/SC16 — Open System Interconnection). Opracowywanie dokumentów przebiega przez cztery fazy generujące odpowiednie rodzaje dokumentów:

- dokumenty robocze (WD — Working Documents)
- propozycje projektu (DP — Draft Proposal)
- projekty standardu międzynarodowego (DIS — Draft International Standards)
- standardy międzynarodowe (IS — International Standards).

Aktualny stan prac oraz duże już, prawie roczne, opóźnienia w stosunku do harmonogramu pozwalają stwierdzić, że standardy międzynarodowe usług i protokołów sesji, prezentacji, transferu zbiorów itp. pojawią się dopiero w latach 1987—1989.

W CCITT działają dwie grupy studialne. Grupa VII (Public Data Network) opracowuje standardy dla modelu ISO/OSI i ma dwie podgrupy zajmujące się warstwą transportową i warstwą sesji. Grupa VIII (Telematic Services) opracowuje standardy warstwy transportowej i sesji dla usług systemu TELETEX.

Należy podkreślić, że materiały ISO i CCITT, dzięki wspólnej dyskusji na poziomie grup roboczych obu organizacji, nie są rozbieżne.

Tabela 2. Zestawienie ważniejszych dokumentów ISO i CCITT, dotyczących wyższych warstw modelu ISO/OSI

Usługa lub protokół	ISO	CCITT
Protokół wymiany dokumentów		T.73
Usługi warstwy sesji	DIS 8326	X.215
Protokół warstwy sesji	DIS 8327	X.225
Usługi transportowe	DIS 8072	X.214
Protokół transportowy	DIS 8073	X.224
Usługi sieciowe	DIS 8348	X.213

Tabela 2 zawiera zestawienie dotychczas wydanych ważniejszych dokumentów ISO i CCITT, dotyczących wyższych warstw modelu ISO/OSI.

Warto również wspomnieć o pracach ECMA (European Computer Manufacturers Association), gdzie w ramach dwóch komitetów technicznych (TC23, TC24) opracowuje się standardy dla warstwy transportowej (ECMA72) i sesji (ECMA75). Ustalenia w ECMA wyprzedzają zwykle prace ISO i CCITT oraz wywierają zauważalny wpływ na kształt standardów międzynarodowych. Podobnie kraje zaawansowane w wykorzystywaniu sieci komputerowych tworzą własne standardy. Przykładem może tu być seria „kolorowych ksiąg", opracowana w Wielkiej Brytanii w National Physical Laboratory i szeroko stosowana w brytyjskiej akademickiej sieci komputerowej JANET.

* * *

Problematyka projektowania i budowy, a zwłaszcza zastosowań sieci komputerowych w Polsce jest opóźniona w stosunku do krajów przodujących w tej dziedzinie niemal o tyle lat, ile ich liczy historia rozwoju sieci. Pierwsze przedsięwzięcie w tym zakresie zostało podjęte w 1979 roku, w ramach problemu resortowego MNSzWiT pod hasłem „Komputeryzacja szkół wyższych", koordynowanego przez Politechnikę Wrocławską.

Jego celem było zbudowanie eksperymentalnej Międzyuczelnianej Sieci Komputerowej (MSK), która po fazie badań i eksperymentów miała stać się zalążkiem trwale funkcjonującej sieci łączącej akademickie środowiska w kraju.

Podstawowymi założeniami konstrukcyjnymi sieci było przyjęcie architektury systemów otwartych ISO/OSI oraz uznanych międzynarodowych standardów, zaś w obszarze nieogarniętym standaryzacją, oparcie się na rozwiązaniach krajów przodujących w dziedzinie sieci.

Podstawowe założenia realizacyjne sieci MSK były następujące:

- sprzęt komputerowy oraz wyposażenie pomocnicze powinny być produkcji krajowej lub krajów RWPG
- całe oprogramowanie powinno być zaprojektowane i uruchomione w kraju.

Obecnie (luty 1985) od ponad roku funkcjonuje homogeniczna wersja sieci ASK, łącząca komputery obliczeniowe ODRA 1305 rozmieszczone w Politechnice Wrocławskiej, Politechnice Śląskiej w Gliwicach oraz w Instytucie Podstaw Informatyki PAN w Warszawie. Heterogeniczna wersja sieci rozpocznie funkcjonowanie w połowie bieżącego roku, po dołączeniu komputera R-32.

Usługi, jakich obecnie dostarcza sieć, są na poziomie przetwarzania skupionego. Po wprowadzeniu wersji heterogenicznej, która wiąże się także z pewnym rozszerzeniem funkcji usługowych, zostanie osiągnięty poziom przetwarzania rozproszonego (tylko dla komputerów ODRA 1305).

Przyjęto lub specjalnie opracowano dla sieci MSK następujące protokoły:

- **poziom fizyczny:** X.21 bis
- **poziom liniowy:** X.25.2
- **poziom sieciowy:** X.25.3, bez usługi datagramowej, jako interfejs do warstwy (protokół zewnętrzny), oraz zmodyfikowana, semidatagramowa wersja X.25 (protokół wewnętrzny) do komunikacji pomiędzy węzłami
- **poziom transportowy:** opracowanie własne na podstawie protokołu ISO/DP N1169, zbliżonego do późniejszego dokumentu ISO/DIS 8073
- **protokół transferu zbiorów:** opracowanie własne oparte na protokole brytyjskim NIFTP (Network Independent File Transfer Protocol), równoważnym protokołowi INWG86 opracowanemu przez IFIP
- **protokół transferu i manipulacji zadań:** opracowanie własne, oparte na protokole brytyjskim NIJTMP (Network Independent Job Transfer and Manipulation Protocol)
- **protokół wirtualnego terminala:** opracowania własne oparte na materiałach roboczych ISO oraz ECMA.

Szczegółowe informacje o strukturze, sprzęcie i oprogramowaniu, a także o doświadczeniach z budowy i dotychczasowej eksploatacji sieci MSK zostaną opublikowane w następnych numerach INFORMATYKI.

LITERATURA

[1] Battek J. i inni: Program budowy Międzyuczelnianej Sieci Komputerowej w latach 1980—1985. CO PWr., marzec 1980
[2] Bernardyn J. i inni: Komputery obliczeniowe w sieci MSK — projekt oprogramowania systemowego. CO PWr., Raport SPR nr 14, 1983
[3] Bilski E.: Projekt techniczno-funkcjonalny MSK — zasady techniczne. ICT i CO PWr., Raport SPR nr 33, 1982
[4] Butrimienko A. W.: Projektowanie i eksploatacja sieci komputerowych, WNT 1983
[5] Praca zbiorowa: Zagadnienia oprogramowania węzła podsieci sieci MSK. Prace Naukowe CO PWr., Studia i Materiały nr 2, 1983.

STANISŁAW WALIGÓRSKI
Instytut Informatyki
Uniwersytet Warszawski

LOGO (3)

Porównanie dialektów APPLE, SINCLAIR i COMMODORE 64

Obserwując rozpowszechnianie się różnych wersji LOGO w Polsce można stwierdzić, że najpopularniejsza staje się wersja na Sinclair ZX Spectrum, ze względu na znaczną liczbę tych mikrokomputerów w kraju, następnie LOGO na Apple II oraz LOGO na Commodore 64. Ta ostatnia wersja jest wzorowana na Terrapin LOGO, dialekcie, w którym pisano wszystkie dotychczasowe przykłady w tym artykule. Natomiast wersja Sinclair LOGO jest wzorowana na Apple LOGO i różni się od niej tylko drobnymi szczegółami. Z kolei, Terrapin i Apple LOGO różnią się stosunkowo niewiele i opanowawszy jedną z tych wersji można bardzo łatwo nauczyć się drugiej.

Zbadamy bliżej podobieństwa i różnice tych trzech dialektów. Następujące komendy i operacje, wprowadzone w dwu poprzednich częściach artykułu są wspólne dla wszystkich trzech wersji: FORWARD, BACK, RIGHT, LEFT, PENUP, PENDOWN, SHOWTURTLE, HIDETURTLE, PENERASE, HOME, SETX, SETY, SETHEADING, XCOR, YCOR, HEADING, WRAP.

Komendy związane z grafiką, różne w Terrapin i Apple LOGO, ale jednakowe w Apple i Sinclair LOGO są następujące

Terrapin	Apple i Sinclair
DRAW	CLEARSCREEN
NODRAW	TEXTSCREEN
CLEARSCREEN	CLEAN
PENCOLOR	SETPC
BACKGROUND	SETBG
SETXY	SETPOS
TOWARDS	TOWARDS
NOWRAP	FENCE

W Apple i Sinclair LOGO BACKGROUND jest operacją udostępniającą numer koloru tła. Podobnie, operacja PENCOLOR (w Sinclair LOGO PENCOLOUR) udostępnia numer koloru kreski rysowanej przez pióro. Procedury SETPOS i TOWARDS w Apple i Sinclair LOGO są jednoargumentowe. Ich argument musi być listą dwuelementową, której elementy stanowią wartości obu współrzędnych punktu, o który chodzi. Ich odpowiedniki w Terrapin LOGO są dwuargumentowe.

Oprócz operacji WRAP i FENCE, w Apple i Sinclair LOGO jest trzeci tryb, WINDOW, którego nie ma w Terrapin LOGO. W trybie tym żółw może poruszać się po płaszczyźnie poza ekranem, który jest tylko oknem, dającym wgląd w otoczenie początku układu współrzędnych.

Działania na liczbach, słowach i listach, wspólne dla wszystkich trzech wersji są następujące: SQRT, SIN, COS, RANDOM, ROUND, REMAINDER, WORD, THING, TEXT, SENTENCE, LIST, FIRST, LAST, BUTFIRST, BUTLAST, PRINT, MAKE.

Ponadto identyczne są wszystkie symbole operacji +, —, *, /, występujące w wyrażeniach, symbole relacji =, <, > a także: **oraz** ". Warto wymienić jeszcze inne komendy, wspólne dla wszystkich trzech dialektów, które nie były dotychczas wprowadzone, a mogą być dość często używane:
FPUT — umieść wartość pierwszego argumentu na pierwszym miejscu listy będącej wartością drugiego argumentu (poprzedni pierwszy element tej listy staje się drugim itd.)
LPUT — umieść wartość pierwszego argumentu na ostatnim miejscu listy będącej wartością drugiego argumentu
RUN — wykonaj działania opisane przez kolejne elementy listy będącej argumentem tej procedury
CATALOG — podaj spis plików na dyskietce lub innym nośniku zewnętrznym (nie działa dla pamięci kasetowej)
LOAD — odczytaj plik o zadanej nazwie
SAVE — zapamiętaj na pliku o podanej nazwie
ERASEFILE — skasuj plik o podanej nazwie (nie działa dla pamięci kasetowej)
ERASE — skasuj procedurę o podanej nazwie
EDIT — przejdź do redagowania procedury o podanej nazwie.

Działania na liczbach, słowach i listach, oznaczone w Terrapin LOGO inaczej niż w pozostałych dwu dialektach są następujące:

Terrapin	Apple i Sinclair
NUMBER?	NUMBERP
WORD?	WORDP
LIST?	LISTP
INTEGER	INT
ATAN	ARCTAN
PRINT1	TYPE
REQUEST	READLIST
READCHARACTER	READCHAR
RC?	KEYP

Ostatnie dwie pozycje tego wykazu oznaczają czytanie pojedynczego znaku oraz badanie, czy naciśnięto klawisz: jeśli tak, to wartością predykatu KEYP jest TRUE, w przeciwnym razie FALSE.

Operacja QUOTIENT istnieje w Apple LOGO i ma to samo znaczenie, co w wersji Terrapin, ale nie istnieje w Sinclair LOGO. W Sinclair LOGO można używać także nazw SINE i COSINE zamiast SIN i COS, a ponadto istnieją w nim procedury TANGENT czyli TAN, COTANGENT czyli COT, ARCSIN, ARCCOS i ARCCOT.

Odpowiednikiem operacji THING? z Terrapin LOGO jest operacją THINGP w Apple LOGO i NAMEP w Sinclair LOGO. Słowa: REPEAT, TO, END, OUTPUT, STOP —

mają jednakowe znaczenie we wszystkich trzech dialektach. W Apple i Sinclair LOGO nie używa się konstrukcji **IF ... THEN ...** ani **IF ... THEN... ELSE**, ale **IF ... [...]** albo **IF ... [...] [...]**. W nawiasach kwadratowych podaje się listę opisów działań do wykonania, podobnie jak w operacji REPEAT.

Skróty

Zamiast pełnych nazw można w pewnych przypadkach stosować skróty. Na ogół słowa skraca się tak samo we wszystkich wersjach, niezależnie od tego, że znaczenie słowa (na przykład BACKGROUND) może być w różnych dialektach różne:

FORWARD	FD	
BACK	BK	
RIGHT	RT	
LEFT	LT	
PENUP	PU	
PENDOWN	PD	
PENERASE	PE	
SHOWTURTLE	ST	
HIDETURTLE	HT	
SETHEADING	SETH	
CLEARSCREEN	CS	
BACKGROUND	BG	
PENCOLOR	PC	
PENCOLOUR	PC	(Sinclair LOGO)
SETBORDER	SETBR	(Sinclair LOGO)
SENTENCE	SE	
BUTFIRST	BF	
BUTLAST	BL	
REQUEST	RQ	(Terrapin LOGO)
READLINE	RL	(Apple i Sinclair LOGO)
READCHAR	RC	(Apple i Sinclair LOGO)
READCHARACTER	RC	(Terrapin LOGO)
PRINT	PR	
EDIT	ED	
OUTPUT	OP	

Ponieważ w Sinclair Spectrum jest wydzielone obrzeże ekranu, które może mieć inny kolor niż jego pozostałość, Sinclair LOGO ma osobną procedurę nadającą ten kolor, SETBORDER, która nie występuje w pozostałych wersjach.

W każdej wersji LOGO można pisać wiele komend i operacji w jednym wierszu, przy czym wiersz LOGO ma długość większą niż jeden wiersz ekranu: jednym ciągiem można zapisać ok. 3,5 wiersza ekranu jako jeden wiersz LOGO. Stosowanie skrótów czyni zapis znacznie bardziej zwartym i jest dogodne zwłaszcza wtedy, gdy korzysta się z list, takich jak w operacji REPEAT lub zdaniach warunkowych.

Edytor Apple i Terrapin LOGO

Procedura redagująca EDIT jest w obu tych wersjach prawie dokładnie taka sama. Po przejściu do trybu poprawiania na ekranie ukazuje się treść poprawianej procedury i migający kursor. Aby poprawić zapis, trzeba doprowadzić kursor do właściwego miejsca tekstu, a następnie wprowadzić odpowiednie zmiany. Uzyskuje się to przez jednoczesne naciskanie klawisza CTRL i jednego z podanych niżej klawiszy literowych:

CTRL-F — przesuń kursor o jedno miejsce w przód
CTRL-B — przesuń kursor o jedno miejsce wstecz
CTRL-A — przesuń kursor na początek wiersza
CTRL-E — przesuń kursor na koniec wiersza
CTRL-N — przesuń kursor do następnego wiersza
CTRL-P — przesuń kursor do poprzedniego wiersza
CTRL-O — rozetnij wiersz, przerzucając część na prawo od kursora do nowego wiersza
CTRL-D — usuń znak, na którym stoi kursor
CTRL-K — usuń część wiersza na prawo od kursora
CTRL-Y (tylko Apple LOGO) — wstaw to, co zostało ostatnio usunięte przez CTRL-K
CTRL-L — przesuń tekst tak, aby kursor znalazł się w rejonie środka ekranu
CTRL-C — zakończ redagowanie
CTRL-G — przerwij redagowanie, nie uwzględniając poprawek.

Naciśnięcie klawisza znakowego (bez CTRL) powoduje wpisanie tego znaku na miejscu kursora. Naciśnięcie klawisza wprowadzania wiersza (ENTER, RETURN albo CR, zależnie od klawiatury) powoduje przejście do nowego wiersza.

Po wyjściu z edytora przez naciśnięcie klawisza CTRL-C, na ekranie ukazuje się informacja, że procedura została zdefiniowana. Treść komunikatu jest taka sama, jak po zdefiniowaniu tej procedury przy użyciu konstrukcji **TO ... END.**

Inne zastosowania klawiszy specjalnych w Apple i Terrapin LOGO

Poza edytorem, klawisze specjalne mogą być używane do poprawiania wpisywanego wiersza, wstrzymywania, zawieszania lub przerywania wykonania komend albo do zmiany zawartości ekranu. W trybie SPLITSCREEN można oglądać rysunek na całym ekranie, z wyjątkiem czterech dolnych wierszy, w których widać to, co jest wprowadzane z klawiatury. Wpisywanie nowych wierszy powoduje przesuwanie się poprzednio wyświetlanych do góry, tak że stają się niewidoczne po wsunięciu się pod część ekranu, zajętą przez rysunek. Podobnie, część rysunku wchodząca w obszar przeznaczony na dolne cztery wiersze tekstu jest niewidoczna. Można jednak, naciskając odpowiednie klawisze, zobaczyć tekst na całym ekranie (razem z częścią zasłanianą przez rysunek) albo zobaczyć rysunek na całym ekranie, przy czym każda z tych operacji jest odwracalna i nie niszczy ani rysunku, ani tekstu. Poszczególne klawisze mają następujące znaczenie:

CTRL-T — pokaż tekst na całym ekranie, czyniąc rysunek niewidocznym
CTRL-S — pokaż cztery dolne wiersze tekstu i rysunek na pozostałej górnej części ekranu
CTRL-L (CTRL-F w Terrapin LOGO) — pokaż rysunek na całym ekranie, czyniąc tekst niewidocznym
CTRL-W — zawieś wykonanie komendy lub przesuwanie się tekstu wyprowadzanego na ekran; wznowienie wykonania komendy lub wyprowadzania tekstu nastąpi po naciśnięciu dowolnego klawisza
CTRL-Z — wstrzymaj wykonywanie komendy; po wstrzymaniu można wykonywać dowolną sekwencję czynności; wznowienie wykonania wstrzymanej komendy następuje po komendzie CO (w Terrapin LOGO można także napisać CONTINUE)
CTRL-G — przerwij wykonanie komendy, operacji lub procedury
CTRL-D — skasuj znak, na którym stoi kursor
CTRL-K — skasuj część wiersza na prawo od kursora
CTRL-A — przerzuć kursor na początek wiersza
CTRL-E — przerzuć kursor na koniec wiersza
CTRL-P — powtórz ostatnio napisany wiersz, o ile nie zawiera REPEAT ani RUN
CTRL-Q — potraktuj następny wprowadzony znak dosłownie, odbierając mu specjalne znaczenie; jeśli je ma; przykładowo, nawias nie będzie potraktowany jako nawias rozpoczynający listę; na ekranie CTRL-Q jest widoczne jako /.

W Apple LOGO istnieją ponadto polecenia:

CTRL-Y — wstaw to, co zostało usunięte ostatnio przez CTRL-K
CTRL-B — przesuń kursor o jedno miejsce wstecz.

Edytor Sinclair LOGO

Komputery Sinclair Spectrum mają wiele trybów wprowadzania znaków z klawiatury. Klawisz CAPS SHIFT włącza pisanie dużych liter, a dla cyfr 5—8 i 0 strzałki we wszystkich czterech kierunkach oraz DELETE (kasowanie). Tryb EXTENDED MODE włącza się specjalnym klawiszem, a jeśli takiego nie ma (na Spectrum z plastykową klawiaturą), przez jednoczesne naciśnięcie klawiszy CAPS SHIFT i SYMBOL SHIFT. Włączenie tego trybu jest sygnalizowane literą E w prawym dolnym rogu ekranu. Po naciśnięciu dowolnego klawisza znakowego następuje automatyczne przejście do trybu normalnego.

Po przejściu do redagowania procedury, na ekranie pojawia się migający kursor, który trzeba doprowadzić do poprawianego miejsca. Poprawiany tekst jest automatycznie podzielony na strony, a poza tym może być przewijany, tak aby można oglądać kolejne wiersze długiego tekstu, stronę po stronie. Do wykonania tych operacji służą następujące klawisze:

CAPS SHIFT 8 (strzałka w prawo) — przesuń kursor o jedno miejsce w przód
CAPS SHIFT 5 (strzałka w lewo) — przesuń kursor o jedno miejsce wstecz

dokończenie na str. 21

Dzięki firmom polonijnym i rzemieślniczym zwiększa się na polskim rynku liczba użytkowanych mikrokomputerów. Chociaż firmy oferujące sprzęt do zastosowań profesjonalnych gwarantują serwis w terminie do 48 godzin od chwili zgłoszenia uszkodzenia, to jednak gdy utrzyma się tempo instalowania nowego sprzętu i równocześnie przybędzie niefachowych użytkowników, mogą powstać spore problemy. Odbiorców sprzętu z państwowych wytwórni też nie możemy skwitować krótkim: sami są sobie winni. Wreszcie niektóre „groźne" awarie często wynikają z braku doświadczenia użytkownika. Być może zamieszczony tekst zaoszczędzi tego rodzaju kłopotów przynajmniej niektórym naszym Czytelnikom.

Mikrokomputerowa pierwsza pomoc

Człowiek jest omylny, jednak skutki jego błędów mogą okazać się niewielkie w porównaniu do tych, jakie może spowodować „pomylony" komputer. O ile człowiek myli się zazwyczaj wykonując obliczenia, o tyle uszkodzenie komputera może prowadzić do utraty ogromnych ilości cennych informacji i to niemal w mgnieniu oka. Z tego właśnie powodu powszechnie zaleca się możliwie częste sporządzanie kopii zbiorów roboczych. Ta pozornie mało twórcza czynność powinna być powtarzana przynajmniej po każdym dniu pracy na komputerze.

Mikrokomputery to w sumie dosyć skomplikowane maszyny i zazwyczaj psują się w wyrafinowany sposób. Dlatego też trudno sporządzić wyczerpującą listę możliwych usterek. W poniższym tekście spróbujemy wspomnieć o kilku najbardziej popularnych. Nie będziemy oczywiście pisać w tym miejscu jak zastąpić serwis komputerowy, jednak w niektórych sytuacjach faktycznie możemy sobie sami poradzić lub przynajmniej skonkretyzować nasze „pretensje" w stosunku do komputera. To ostatnie jest o tyle ważne, że z chwilą przybycia serwisu komputery mają zwyczaj złośliwie udawać, że wszystko jest w porządku. Sporządzona na drukarce kopia naszych działań może okazać się bardzo pomocna w zlokalizowaniu uszkodzenia. Nie chodzi tu tylko o zaoszczędzenie czasu osoby naprawiającej, ale też w dużej mierze o zaoszczędzenie naszych nerwów przy wielokrotnym wzywaniu serwisu do tego samego uszkodzenia.

Niestety dosyć często zdarza się, że fabrycznie nowe komputery nie funkcjonują poprawnie. Dlatego też warto upierać się, aby instalacja i sprawdzenie systemu (zazwyczaj wpisywane do rachunku) faktycznie zostało wykonane. Nowemu komputerowi warto na początku poświęcić trochę czasu, choćby wprowadzając przykładowe procedury zamieszczane w instrukcji obsługi. Nie należy przy tym przeceniać własnego braku doświadczenia. Jeśli uważnie przeczytaliśmy komentarz do programu i mamy pewność, że wprowadzony program jest identyczny z tym, co wydrukowano w instrukcji, a mimo to nie osiągamy spodziewanego rezultatu, to istnieje duże prawdopodobieństwo, że system nie funkcjonuje prawidłowo.

Zakłócenia elektrostatyczne nie są na szczęście zbyt często występującą bolączką. Jeżeli jednak po kilkukrotnym przejściu pokoju, w którym pracuje komputer, powodujemy wyładowanie elektrostatyczne w zetknięciu z kaloryferem, to możemy spodziewać się kłopotów. Najczęściej objawiają się one nieoczekiwaną zmianą wyświetlanej na monitorze informacji lub też utratą kontroli nad systemem. Zazwyczaj wyzerowanie systemu (ang. RESET) przywraca mu pełnię sił witalnych. Środki zaradcze mogą być najróżniejsze: poczynając od zmiany umeblowania (likwidacja plastikowej wykładziny, inny stół etc.), a kończąc na zmianie stroju operatora, który powinien unikać ubiorów z włókien sztucznych.

Zakłócenia napięcia zasilającego

Różne typy komputerów mają różną odporność na niespodzianki z gniazdka sieciowego. Komputerom rodem z Wielkiej Brytanii często nie podoba się, że w sieci zamiast przepisowych 220 V jest niewiele ponad 200. Krótkie nawet przerwy w napięciu zasilania zazwyczaj kończą się utratą kontaktu z komputerem lub przynoszą efekt identyczny jak zerowanie. Osobną klasą zakłóceń są przepięcia generowane przez innych odbiorców energii elektrycznej. Należy oczekiwać, że znakomita większość Czytelników potrafi przewidzieć konsekwencje zasilania z jednego gniazdka komputera i grzejnika z termostatem. Jeżeli jednak komputer musi pracować w „uprzemysłowionej" okolicy, to należy pomyśleć o kupnie dobrego stabilizatora. Jak wskazuje jego nazwa powinien on także poradzić sobie ze zbyt niskim napięciem w sieci. Jeżeli mimo stabilizatora zakłócenia nadal wyprowadzają komputer z równowagi, to mamy dwie możliwości: poszukać innego typu komputera, z lepiej rozwiązanym zasilaczem lub też zamienić pomieszczenie w puszkę Faraday'a. To drugie rozwiązanie, choć kosztowne, często bywa mało skuteczne.

Problem nie włączonej drukarki

Proszę się nie śmiać, tego typu błąd zdarza się wcale nie tak rzadko nawet wieloletnim użytkownikom komputerów. Bardzo często zdarza się, że serwis przyjeżdża, aby stwierdzić... że skończył się założony na drukarkę papier. Zazwyczaj gdy komputer dochodzi do fragmentu programu realizującego wydruk, który z jakiś powodów nie może zostać wykonany, to potulnie czeka aż operator włączy ją do sieci, przełączy w tryb „on line" lub wetknie kabel połączeniowy.

Z problemem tym można sobie stosunkowo łatwo poradzić nawet wtedy, gdy interfejs z drukarką jest maksymalnie uproszczony. Należy przed każdą procedurą przesyłającą dane na drukarkę umieścić krótki program. Powinien on wyświetlać na ekranie monitora tekst: „Włącz drukarkę, ośle!", a następnie wysyłać na drukarkę kod przejścia do nowej linii (LF — w języku BASIC instrukcją LPRINT bez parametrów). W następnej linii programu należy zamieścić polecenie wymazania napisu na monitorze i dalej sterowanie można już przekazać do procedury drukowania. W ten prosty sposób, ilekroć „program „zawiesi się" na niegotowej drukarce, dostaniemy od komputera odpowiednią reprymendę. Gdy drukarka będzie gotowa napis będzie zbyt krótko na ekranie, aby można było cokolwiek odczytać.

Uszkodzenia monitora

Typowe uszkodzenia to „pływanie" obrazu lub brak odchylania (pozioma jasna kreska). Całkowity brak obrazu jest uszkodzeniem stosunkowo rzadkim i z reguły poprzedzonym gwałtownym rozbłyśnięciem, trzaskiem lub obłoczkiem dymu. Brak obrazu najczęściej spowodowany jest złym połączeniem z komputerem lub też awarią samego komputera. Uszkodzony lub niepodłączony monitor zazwyczaj nie blokuje realizacji programu. W mikrokomputerach wyższej klasy, wyposażonych w drukarkę możemy przeprowadzić stosunkowo prosty test: należy ściągnąć system z dyskietki (MS-DOS, PC-DOS lub CP/M), wprowadzić z klawiatury CTRL-P i podać przykładowo komendę DIR. Wydruk znajdujących się na dyskietce programów będzie sugerował, że komputer jest w porządku.

Uszkodzenia pamięci RAM

Uszkodzenia tego typu objawiają się zazwyczaj od początku użytkowania mikrokomputera lub zaraz po dołączeniu bloku rozszerzającego obszar pamięci. Niektóre układy pamięci zaczynają wadliwie funkcjonować w podwyższonej temperaturze, a więc dopiero po pewnym okre-

sie pracy komputera. Błędy pamięci bywają często trudne do wykrycia. Częstym symptomem jest fakt, że niektóre programy pracują prawidłowo, podczas gdy inne „padają" od początku lub powodują zawieszenie systemu w trakcie realizacji. Jeżeli uszkodzenie pojawi się w obszarze przeznaczonym na pamięć obrazu, to zazwyczaj widać je w postaci ciemnych punktów lub dziwnych zjaw na ekranie. Jeżeli uszkodzenie trafiło się w obszarze, w którym przechowywane są ważne zmienne systemowe, to po wyzerowaniu komputera na całym ekranie pojawią się dziwaczne stwory, a w tle czasem paski lub nawet krata. W tym przypadku uszkodzenie nie budzi wątpliwości — objawy złego funkcjonowania są „stabilne".

Najwięcej kłopotu sprawiają uszkodzenia pojedynczych bitów pamięci RAM w obszarze, w który ładowany jest system operacyjny lub program użytkowy. Przez stosunkowo długi czas uszkodzenie może zupełnie nie objawiać się, gdyż akurat nie było odwołań do fragmentu kodu znajdującego się w tym miejscu. Jeśli jednak już natrafimy na „chore" miejsce, to prawie na pewno zakończy się to utratą kontaktu z komputerem. Wyświetlany na ekranie obraz może się zmienić lub nie, ale komputer nie będzie reagował na przyciskane klawisze.

Do wykrywania błędów tego typu służą specjalne programy nazywane testami pamięci. Najprostszy test można napisać posługując się nawet językiem BASIC. Należy najpierw instrukcją POKE zapisać zerami cały obszar pamięci, przeznaczony na programy użytkownika (z wyjątkiem miejsca, gdzie znajduje się sam program testu pamięci), a następnie odczytać wszystkie lokacje, sprawdzając, czy mają wartość zero. Potem należy zapisać pamięć wartością 255 (OFF_H) i wykonać sprawdzenie. Całość warto powtórzyć kilka razy, mimo że czas wykonywania tego typu programu jest stosunkowo długi. Jeżeli nie dysponujemy interpretem języka BASIC lub innym programem narzędziowym, który pozwoliłby nam zrealizować opisany algorytm (w komputerach zorientowanych na określone zastosowania, np. biurowych, jest to całkiem prawdopodobne!), możemy posłużyć się inną metodą. Zadowalające wyniki może dać program elektronicznego formularza rachunkowego typu VisiCalc. Należy po prostu zdefiniować poszczególne pola w ten sposób, aby w każdej komórce była wykonywana operacja polegająca na zwiększeniu o jeden wartości z poprzedniej komórki. Kolejne pola należy definiować aż do wyczerpania przeznaczonego na ten cel obszaru pamięci RAM. Gdy proces definiowania zostanie zakończony, należy ponawiać zlecenie przeliczenia wartości dopóki komputer nie „zatka się". Przykładowo dla APPLE i programu VisiCalc przeliczenie może być wykonane 32 razy. Jeżeli w obszarze wykorzystywanym przez program istnieje uszkodzenie, to końcowy wynik kalkulacji nie będzie prawidłowy. Wykonanie pojedynczego przeliczenia nie gwarantuje jeszcze, że uszkodzony bit nie będzie przypadkowo ustawiony na wartość, jaką powinien mieć faktycznie. Wielokrotne powtórzenie przeliczania zmniejsza prawdopodobieństwo takiego przypadku niemal do zera.

Bezsprzeczną wadą badania pamięci poprzez niekonwencjonalne wykorzystanie programu elektronicznego formularza rachunkowego jest kłopot z określeniem lokacji, która uległa uszkodzeniu. Opisany wcześniej algorytm prostego programu testującego pamięć można bez trudu przystosować tak, aby wyświetlał adres komórki, w której stwierdzona została nieprawidłowość.

Pozostał nam jeszcze jeden element systemu, wyjątkowo podatny na uszkodzenia. Chodzi tu o pamięć masową na dyskach elastycznych. Ale o tym — innym razem.

AJP

Na komfort pracy z mikrokomputerem składa się między innymi szybkość, z jaką realizuje on nasze zlecenia. Cytowane w tekście testy ograniczają się niestety tylko do pewnego wycinka rzeczywistości, jakim jest interpreter języka BASIC. Podobne testy można by wymyślić i dla innych języków — jednak mikrokomputery wyposażone są „firmowo" zazwyczaj tylko w BASIC. Interpretery lub kompilatory innych języków to najczęściej oferta niezależnych producentów oprogramowania, co najwyżej przystosowana dla danego typu mikrokomputera. Uniwersalnych metod testowania innych cech komputerów (np. szybkości współpracy z pamięcią masową) na razie nie opracowano i raczej nieprędko to nastąpi. Wynika to ze znacznej liczby różnych sposobów realizacji typowych wydawałoby się — funkcji.

Siedem z cytowanych w artykule testów publikowaliśmy już w zeszłym roku. Wielu Czytelników zwróciło nam jednak uwagę, że znacznie częściej wykorzystywany jest zestaw złożony z ośmiu testów. Ponieważ ósmy test dotyczy stosunkowo ważkiej cechy — szybkości wyliczania funkcji, zdecydowaliśmy się na jego publikację. Mając na uwadze nowych Czytelników i znane trudności ze zdobyciem INFORMATYKI, drukujemy cały zestaw.

Uwaga: autor poniższego artykułu przetłumaczył z języka angielskiego na język polski m.in. komentarz umieszczany w linii 100. W oryginale użyte jest słowo „BENCHMARK" — nie ma to jednak praktycznego wpływu na mierzony czas. Początkującym Czytelnikom zwracamy uwagę, że niektóre interpretery języka BASIC wymagają pełnego zapisu po słowie kluczowym THEN. Numer linii należy wtedy poprzedzić instrukcją GOTO.

Wyścig z czasem

Brytyjski miesięcznik PERSONAL COMPUTER WORLD (PCW) regularnie publikuje omówienia nowo pojawiających się na rynku mikrokomputerów. Jedną z badanych cech jest szybkość wykonywania podstawowych instrukcji języka BASIC. PCW stosuje w tym celu zestaw ośmiu krótkich programów testowych — ten sam od założenia pisma, tj. od 1978 roku. Umożliwia to porównywanie wyników testów dla różnych komputerów.

Styczniowy numer tego czasopisma przynosi bieżącą listę 68 mikrokomputerów o najszybszych interpreterach języka BASIC. Pierwsze miejsce na tej liście zajmuje — już od dwóch lat — rewelacyjny **SAGE II**, znany też z oryginalnej metody, jaką zastosowano przy jego projektowaniu. Na drugim miejscu jest mniej znany i znacznie nowszy **CRYSTAL**. Oba te urządzenia zawierają mikroprocesor MOTOROLA 68000 oraz, co chyba w tym wypadku ważniejsze, realizują częściowo skomplikowaną wersję języka BASIC, tzw. kod pośredni. Nieznacznie wolniejszy jest trzeci na liście **IBM AT**, zbudowany na nowym mikroprocesorze INTEL 80286.

Uznawany za największego konkurenta rodziny IBM PC — **APPLE MACINTOSH** zajmuje dopiero 12 miejsce, jednakże głównie ze względu na wyjątkowo wolną, jak na tę klasę maszyn, realizację funkcji. Jeśli pominiemy wyniki testu nr 8, pozwalającego ocenić prędkość wyliczania wartości funkcji sinus i logarytm, okaże się, że średnia z pozostałych siedmiu testów jest niemal identyczna, jak dla maszyn IBM AT i OLIVETTI M 24. Oznacza to mniejszą przydatność MACINTOSHA dla wszystkich tych zastosowań, gdzie wykonywane są skomplikowane obliczenia matematyczne, do czego nie był on jednak projektowany [1]). Za to najstarszym na liście i wcale nie ostatnim jest słynny **APPLE II.**

Najszybszym mikrokomputerem 8-bitowym okazał się mało znany **JONOS**, używający również bardzo szybkiego

---

[1]) Sąd autora tekstu jest może nazbyt pochopny. W zagadnieniach numerycznych, choć pożądana jest szybkość, to jednak istotniejsza jest często dokładność numeryczna. Warto dodać, że MACINTOSH zdobył sobie dużo zwolenników na amerykańskich uniwersytetach — raczej nie ze względu na piktogramy i myszkę (przyp. AJP).

mikroprocesora Z80B. Wysoką pozycję zajmuje też trzyletni już BBC B firmy ACORN. Wydaje się, że brytyjskie ministerstwo oświaty dokonało właściwego wyboru zalecając w swoim czasie ten solidny i szybki model dla szkół średnich i podpisując odpowiednią umowę z producentem.

Bardzo szybki jest też coraz popularniejszy AMSTRAD CPC 464. Wydaje się, że może on w najbliższym czasie stać się najpopularniejszym brytyjskim mikrokomputerem domowym. Jest on nie tylko szybki, ale również solidnie wykonany i stosunkowo tani. Dostępnością oprogramowania przewyższa on wszystkich potencjalnych konkurentów. Stosunkowo nisko plasuje się natomiast niedoszły przebój rynkowy sir Clifa Sinclair'a — **QL**, pomimo stosowania mikroprocesora MOTOROLA 68008.

Niestety, brak jest na liście niektórych mikrokomputerów, które z pewnością by się na niej znalazły, gdyby poddano je testowi. Mam tu na myśli takie urządzenia, jak: **ZENITH** (różne modele), **APPLE, LISA, CROMEMCO** (różne modele) czy chociażby popularny **COMMODORE 64** (omawiany w numerze 5, 1983 PCW, lecz nie testowany na szybkość). Z tego względu, jak również z uwagi na ograniczony zakres testów, opublikowanej listy nie można traktować jako w pełni aktualnej, kompletnej i niepodważalnej klasyfikacji.

W tabeli zestawiono 12 początkowych pozycji z listy miesięcznika PCW oraz 14 dalszych, wybranych dla bardziej znanych u nas mikrokomputerów. Dla tych czternastu, po nazwie, podano w nawiasie pozycję zajmowaną na liście. Rok określa w przybliżeniu wiek modelu; jest to rok, w którym testowano urządzenie. Zazwyczaj pojawiało się ono na rynku około pół roku wcześniej.

Na użytek Czytelników, którzy chcieliby przetestować nie figurujące na tej liście mikrokomputery, podajemy zestaw programów testowych stosowanych m.in. prz PCW. Wszystkie są napisane w języku BASIC i każdy z nich zawiera wykonywaną 1000 razy pętlę.

Test 1 zawiera „pustą" instrukcję FOR...NEXT i określa szybkość jej realizacji. **Test 2** zawiera również 1000 pustych pętli, lecz realizowanych poprzez instrukcję skoku. W **teście 3** dodano trochę arytmetyki.
Test 4 realizuje te same instrukcje, lecz zamiast zmiennych w linii 140 użyto stałych.
Test 5 pozwala zmierzyć szybkość realizacji instrukcji wywołania „pustego" podprogramu (GO SUB).
Test 6 miał prawdopodobnie testować szybkość instrukcji DIM, lecz w takim razie nie został wymyślony zbyt szczęśliwie (instrukcja w linii 130 powinna być umieszczona w taki sposób, aby w istotny sposób wpływała na czas realizacji programu) i w zasadzie test 6 stanowi poszerzenie testu 5 o 5000 wykonań pustej pętli FOR. **Test 7** otrzymano z poprzedniego, dodając instrukcję przypisania realizowaną na elemencie małej tablicy. Wreszcie **test 8** pozwala określić szybkość obliczania podstawowych funkcji.

```
100 REM TEST 1
110 PRINT "S"
120 FOR K=1 TO 1000
130 NEXT K
140 PRINT "E"
150 END
```

```
100 REM TEST 2
110 PRINT "S"
120 K=0
130 K=K+1
140 IF K<1000 THEN 130
150 PRINT "E"
160 END
```

```
100 REM TEST 3
110 PRINT "S"
120 K=0
130 K=K+1
140 A=K/K*K+K-K
150 IF K<1000 THEN 130
160 PRINT "E"
170 END
```

```
100 REM TEST 4
110 PRINT "S"
120 K=0
130 K=K+1
140 A=K/2*3+4-5
150 IF K<1000 THEN 130
160 PRINT "E"
170 END
```

Używając tych programów przetestowano zakupiony przez Instytut Ekonometrii Uniwersytetu Łódzkiego mikrokomputer IMP-86, produkowany przez firmę IMPOL II. Jest to model kompatybilny z IBM PC i wykorzystuje oryginalne oprogramowanie IBM. Dla wszystkich testów otrzymaliśmy czasy nieco lepsze niż podane w czasopiśmie PCW dla IBM PC. Interesujące byłoby porównanie wyników testów innych mikrokomputerów produkowanych w kraju, np. MERITUM czy COMPAN.

Lista najszybszych mikrokomputerów według Personal Computer World, January 1985

Typ mikrokomputera	Rok testowania	Czasy realizacji poszczególnych testów (sekundach)								Średnia czasów
		T1	T2	T3	T4	T5	T6	T7	T8	
SAGE II	1983	0,50	0,70	1,30	1,70	2,10	5,10	6,40	18,00	4,47
CRYSTAL	1984	0,70	1,70	4,60	3,90	4,30	7,70	13,60	18,70	6,91
IBM AT	1984	0,80	2,20	4,90	5,10	5,60	9,40	15,00	13,90	7,11
OLIVETTI M24	1984	0,80	2,50	5,20	5,20	5,70	10,00	15,30	16,60	7,66
MAD1	1984	0,70	2,60	5,50	5,70	6,20	10,90	16,80	17,30	8,21
XEROX 16/8	1984	1,20	3,60	7,40	7,70	8,70	16,00	4,70	24,70	9,25
FUTURE FX-20	1983	1,20	3,40	6,80	7,00	7,90	14,50	22,20	17,50	10,06
TULIP	1983	1,00	3,70	6,00	6,10	7,80	15,50	23,20	17,50	10,11
HP 110	1984	0,65	3,40	6,60	6,80	7,70	14,20	21,80	21,90	10,38
JONOS	1984	0,80	2,50	6,70	6,70	7,20	12,70	20,10	34,40	11,39
ADVANCE 86	1984	1,10	3,50	7,50	7,60	8,30	14,90	23,20	26,10	11,53
APPLE MACINTOSH	1984	0,24	0,65	6,00	6,40	7,10	8,60	15,90	52,30	12,15
BBC B (19)	1982	1,00	3,10	8,20	8,70	9,10	13,90	21,40	51,00	14,55
AMSTRAD (20)	1984	1,09	3,28	9,16	9,61	10,20	19.03	30,18	34,20	14,59
SIRIUS 1 (21)	1982	1,80	5,30	10,70	11,10	12,90	24,20	37,10	27,90	16,38
APRICOT (23)	1983	1,60	5,20	10,60	11,00	12,40	22,90	35,40	24,40	16,69
IBM PC (26)	1981	1,50	5,20	12,10	12,60	13,60	23,50	37,40	35,00	17,61
QL (31)	1984	2,10	6,40	10,70	10,30	13,20	26,10	61,80	25,80	19,55
TRS-80 II (36)	1980	1,00	5,00	13,00	13,00	14,00	23,00	35,00	60,00	20,50
IBM PC JUNIOR (39)	1984	1,90	6,40	11,20	15,00	16,50	29,30	46,60	47,40	21,79
APPLE III (42)	1982	1,70	7,20	13,50	14,50	16,00	27,00	42,50	75,00	24,68
VIC-20 (48)	1981	1,40	8,30	15,50	17,10	18,30	27,20	42,70	99,00	28,69
APPLE II (49) ś	1979	1,30	8,50	16,00	17,80	19,10	28,60	44,80	107,00	30,39
ZX-81 (61)	1981	4,50	6,90	16,40	15,80	18,60	49,70	68,50	229,00	51,18
ZX-SPECTRUM (63)	1982	4,80	8,70	21,10	20,40	24,00	55,30	80,70	253,00	58,50
ATARI (65)	1980	2,30	7,40	19,90	23,20	26,80	40,70	61,50	431,00	76,60

```
100 REM TEST 5
110 PRINT "S"
120 K=0
130 K=K+1
140 A=K/2*3+4-5
150 GOSUB 190
160 IF K<1000 THEN 130
170 PRINT "E"
180 END
190 RETURN
```

```
100 REM TEST 6
110 PRINT "S"
120 K=0
130 DIM(5)
140 K=K+1
150 A=K/2*3+4-5
160 GOSUB 220
170 FOR L=1 TO 5
180 NEXT L
190 IF K<1000 THEN 140
200 PRINT "E"
210 END
220 RETURN
```

```
100 REM TEST 7
110 PRINT "S"
120 K=0
130 DIM M(5)
140 K=K+1
150 A=K/2*3+4-5
160 GOSUB 230
170 FOR L=1 TO 5
180 M(L)=A
190 NEXT L
200 IF K<1000 THEN 140
210 PRINT "E"
220 END
230 RETURN
```

```
100 REM TEST 8
110 PRINT "S"
120 K=0
130 K=K+1
140 A=K↑2
150 B=LOG(K)
160 C=SIN(K)
170 IF K<1000 THEN 130
180 PRINT "E"
190 END
```

Przy testach tego rodzaju należy pamiętać, że dotyczą one tylko jednej z cech mikrokomputera, a i to traktowanej fragmentarycznie. Zacytowany zestaw testów np. nie pozwala określić szybkości operacji graficznych na ekranie. Znaczenie danej cechy jest różne dla różnych użytkowników, w zależności od charakteru zastosowań i metod pracy. Gdy w grę wchodzą cechy, których nie obejmują cytowane programy, trzeba niestety samemu wymyśleć test. Jest to zazwyczaj o tyle trudne, że wkraczamy w obszar rozbieżności między różnymi wersjami języka BASIC.

RYSZARD KURZYJAMSKI
Uniwersytet Łódzki

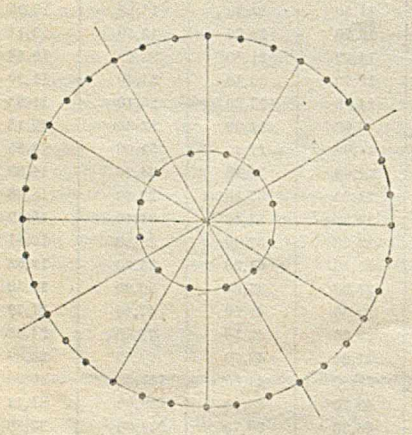
Hasło „metody numeryczne" wywołuje u wielu osób gęsią skórkę. Tymczasem zanim zdecydujemy się na przejście do programowania w języku ASSEMBLER warto czasem zajrzeć do podręcznika matematyki. Może się bowiem nagle okazać, że to, co przedłużało czas realizacji programu niemal w nieskończoność, można zrealizować inną metodą, znacznie szybszą i równie dokładną.

W zasadzie wszyscy wiedzą, że obliczanie funkcji trygonometrycznych trwa dłużej niż mnożenie. A jednak w niemalże wszystkich programach kreślących okręgi — współrzędne kolejnych punktów wyliczane są mozolnie z wykorzystaniem sinusa i cosinusa. Jak pokazuje poniższy tekst można to samo zrobić znacznie szybciej.

Przykładowe programy zostały zapisane w języku BASIC w wersji zaimplementowanej na komputerach AMSTRAD/SCHNEIDER CPC 464. Jednak nikomu nie powinno sprawić wiele kłopotu przeniesienie ich na dowolny inny komputer. AMSTRAD szczególnie nadaje się do zastosowań graficznych dzięki wysokiej rozdzielczości ekranu. Nie ma to jednak wpływu na algorytm przedstawionych procedur.

Sposoby kreślenia okręgów

W przedstawionych poniżej programach procedura kreśląca koło poprzedzona jest fragmentem umożliwiającym wprowadzenie z klawiatury danych, określających współrzędne środka okręgu, promień oraz liczby punktów, które będą wykorzystywane do jego wykreślania. Liczba punktów, wystarczająca do wykreślania okręgu (zakładając maksymalną dokładność przy danej rozdzielczości monitora lub sterownika CRT) związana jest z długością promienia okręgu i zwiększa się wraz z jego długością (rys. 1).



Rys. 1. Liczba punktów składających się w okrąg zależy od jego promienia

Okręgi o mniejszych promieniach można wykreślić wykorzystując mniejszą liczbę punktów. Zbyt duża liczba punktów niepotrzebnie wydłuża czas kreślenia, gdy przekroczony został zakres rozdzielczości [1]).

Przy kreśleniu okręgu zazwyczaj wykorzystywane są równania parametryczne. Dla okręgu o promieniu r i środku s(a,b) kolejne punkty można wyliczyć z następujących wzorów:

$$x_k = a + r \sin(k\Delta\varphi)$$
$$y_k = b + r \cos k\Delta\varphi)$$
$$0 \leqslant k\varphi\Delta \leqslant 360$$

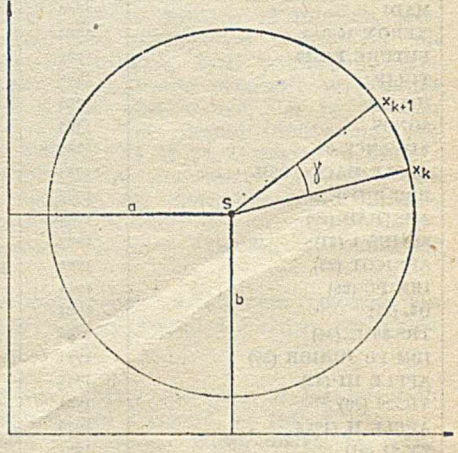
Przykładowy program wykorzystujący tę metodę przedstawia wydruk 1. W tym przykładzie zamiast liczby punktów należy podać kąt $\Delta\varphi$ określający co ile stopni będzie obliczany i rysowany punkt okręgu.

[1] Szczególnie wyraźnie widać to przy zmianie trybu rozdzielczości np. w komputerze AMSTRAD — przyp. AJP.

```
10 REM******************
20 REM
30 REM program 1
40 REM
50 REM******************
60 CLS
70 INPUT "wspolrzedne srodka"; a%, b%
80 INPUT "promien": r%
90 INPUT "kat"; d
100 REM*****************
110 REM
120 REM kreslenie okregu
130 REM
140 REM*****************
150 CLS
160 DEG
170 ORIGIN a%, b%
180 FOR k=0 TO 360 STEP d
190 x%=r%*COS(k) : y%=r%*SIN(k)
200 PLOT x%, y%
210 NEXT
```

Wydruk 1. Program 1

Wykorzystanie w programie równań parametrycznych nie jest rozwiązaniem szczególnie efektywnym. Jakkolwiek program kreślący jest bardzo prosty, wymaga jednak dla każdego punktu okręgu wykonania przez kom-



Rys. 2. Okrąg w układzie współrzędnych. Parametry a i b określają współrzędne środka okręgu s

puter wyliczeń wartości funkcji sinus i cosinus, co powoduje, że kreślenie okręgu odbywa się stosunkowo wolno.

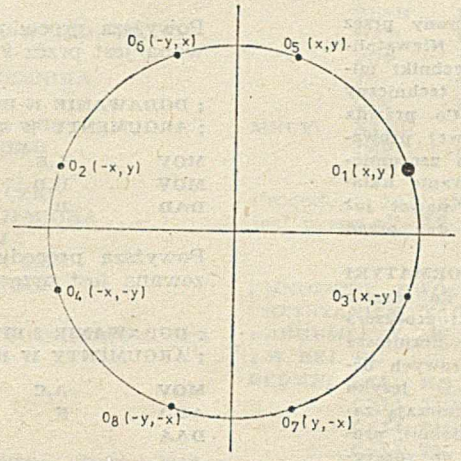
Opierając się na znanych wzorach trygonometrycznych można otrzymać równania, dla których wystarczy tylko raz obliczyć wartości funkcji sinus i cosinus dla kąta γ. Kąt γ otrzymujemy w wyniku podzielenia 360° (pełny okrąg) przez liczbę punktów, które chcemy wykorzystać do wykreślenia okręgu. Kolejne punkty okręgu wyznaczane są iteracyjnie według następujących wzorów (rys. 2).

$$x_{k+1} = a + (x_k - a) \quad \cos\gamma - (y_k - b) \quad \sin\gamma$$
$$y_{k+1} = b + (x_k - b) \quad \sin\gamma + (y_k - a) \quad \cos\gamma$$

Uwaga: sin γ i cos γ — wartości stałe $\gamma = \frac{360}{N}$, gdzie N określa liczbę punktów okręgu. Opisana metoda znacznie przyspiesza kreślenie okręgu. Przykładem wykorzystania tej metody jest PROGRAM 2 (wydruk 2).

```
10 REM******************
20 REM
30 REM program 2
40 REM
50 REM******************
60 CLS
70 INPUT "wspolrzedne srodka"; a%, b%
80 INPUT "promien"; r%
90 INPUT "liczba punktow"; n%
100 REM*****************
110 REM
120 REM kreslenie okregu
130 REM
140 REM*****************
150 CLS
160 DEG
170 ORIGIN a%, b%
180 p=360/n%
190 w1=COS(p):w2=SIN(p)
200 xp=r%:yp=0
210 FOR k%=1 TO n%
220 x=xp*w1—yp*w2
230 y=xp*w2+yp*w1
240 PLOT x, y
250 xp=x:yp=y
260 NEXT
```

Wydruk 2. Program 2



Rys. 3. Dodatkowe siedem punktów, które można uzyskać metodą prostych operacji mając wyliczone współrzędne jednego punktu

```
10 REM******************
20 REM
30 REM program 3
40 REM
50 REM******************
60 CLS
70 INPUT "wspolrzedne srodka"; a%, b%
80 INPUT "promien"; r%
90 INPUT "kat"; d
100 REM*****************
110 REM
120 REM kreslenie okregu
130 REM
140 REM*****************
150 CLS
160 DEG
170 ORIGIN a%, b%
180 n=360/8
190 FOR k=0 TO n STEP d
200 x%=r%*COS(k):y%=r%*SIN(k)
210 PLOT x%, y%
220 PLOT —x%, y%
230 PLOT x%,—y%
240 PLOT —x%,—y%
250 PLOT y%, x%
260 PLOT —y%, x%
270 PLOT y%,—x%
280 PLOT —y%,—x%
290 NEXT
```

Wydruk 3. Program 3

```
10 REM******************
20 REM
30 REM program 4
40 REM
50 REM******************
60 CLS
70 INPUT "wspolrzedne srodka"; a%, b%
80 INPUT "promien"; r%
90 INPUT "liczba punktow"; n%
100 REM*****************
110 REM
120 REM kreslenie okregu
130 REM
140 REM*****************
150 CLS
160 DEG
170 ORIGIN a%, b%
180 p=360/n%
190 m%=n%/8
200 w1=COS(p):w2=SIN(p)
210 xp=r%:yp=0
220 FOR k%=1 TO m%
230 x=xp*w1—yp*w2
240 y=xp*w2+yp*w1
250 xp=x:yp=y
260 x%=x:y%=y
270 PLOT x%, y%
280 PLOT —x%, y%
290 PLOT x%,—y%
300 PLOT —x%,—y%
310 PLOT y%, x%
320 PLOT —y%, x%
330 PLOT y%,—x%
340 PLOT —y%,—x%
350 NEXT
```

Wydruk 4. Program 4

Czas kreślenia okręgu pierwszą lub drugą metodą można jeszcze kilkakrotnie skrócić wykorzystując symetrię okręgu. Wyznaczenie jednego punktu pozwala w bardzo prosty sposób określić dodatkowo współrzędne dalszych siedmiu punktów okręgu, bez konieczności wykonywania dla nich obliczeń funkcji trygonometrycznych (PROGRAM 1) lub operacji mnożenia (PROGRAM 2, rys. 3). Uwzględniając w obliczeniach symetrię okręgu, PROPRAM 1 oraz PROGRAM 2 zmienią się odpowiednio w PROGRAM 3 i PROGRAM 4 (wydruk 3 i 4).

MAREK DORYWALSKI
Przemysłowy Instytut Informatyki
Warszawa

Po okresie prosperity firmy produkujące mikrokomputery przeżywają kryzys. Na rynku angielskim jedynie AMSTRAD wychodzi „na swoje", wypracowując zysk w wysokości 20 milionów funtów. Z ogłoszonych pod koniec roku raportów wynika, że firmy takie jak COMMODORE czy ACORN ponoszą codziennie straty rzędu... miliona funtów. Niewiele lepiej wygląda sytuacja firmy SINCLAIR. Wprawdzie komputerów QL i SPECTRUM+ sprzedaje się dwa razy więcej niż produktów firmy COMMODORE i ACORN razem wziętych, to jednak SINCLAIR zmuszony był zwolnić 20 (spośród 120) pracowników. Sprzedawcy zgodzili się na przejęcie nagromadzonych w magazynach zapasów SPECTRUM+ i QL tylko pod warunkiem opóźnienia promocji nowego modelu komputera SPECTRUM 128 na rynku angielskim aż do lutego 1986. Sir Clive Sinclair — szef firmy — nie mógł jednak oprzeć się pokusie pokazania światu nowego produktu. SPECTRUM 128 składane przez firmę Investronica właśnie pojawiło się w... Hiszpanii. Spodziewają się go też niektóre sklepy w RFN.

Hiszpańskie SPECTRUM zawiera ten sam układ ULA, co dotychczasowy model. Sugeruje to, że możliwości graficzne SPECTRUM 128 nie zostały poprawione. Jest to sprzeczne z plotkami o nowej wersji mikrokomputera. Plotki te wynikły z cech prototypów dostarczonych firmom produkującym oprogramowanie. Jest więc prawdopodobne, że na rynek angielski trafi nieco inna wersja SPECTRUM.

•

Rozczarowani nabywcy komputera AMSTRAD CPC 664 zyskali sprzymierzeńca w firmie DK'tronics, która do tej pory znana była z przystawek do ZX SPECTRUM. Oferuje ona za 50 funtów przystawkę dodającą 64 KB RAM do pamięci CPC 664. Podobnie jak w przypadku AMSTRADA CPC 6128 dodatkowa pamięć nie może być wykorzystywana przez interpreter języka BASIC. Firma DK zapowiedziała też kolejną przystawkę z pamięcią 256 KB za cenę 100 funtów i tzw. dysk na krzemie (pamięć podtrzymywana bateryjnie) również o pojemności 256 KB.

Odpowiednik mikroprocesora 8080 produkowany przez CEMI, znajduje coraz większe zastosowanie. Niewątpliwie należy się cieszyć z upowszechniania techniki mikroprocesorowej, choć szkoda, że postęp techniczny opiera się na tak archaicznym układzie. Co prawda pracownicy Instytutu Technologii Elektronowej prowadzą dyskretny sondaż środowiska, dając do zrozumienia, że niebawem wezmą się za opracowywanie układów bardziej nowoczesnych. Gdyby jednak nawet już dziś zapadły decyzje na tzw. szczeblu, to i tak przed nami jeszcze wiele lat oczekiwania.

Mamy nadzieję, że proces wydawniczy INFORMATYKI nie będzie aż tak długi, toteż oddajemy do druku pierwszą część zestawu procedur dla mikroprocesora 8080. Zdając sobie sprawę z tego, że coraz liczniejsza grupa użytkowników systemów mikroprocesorowych będzie musiała zająć się programowaniem w języku ASSEMBLER (niewielkie możliwości 8080 ograniczają zakres wykorzystywania języków wyższego poziomu), proponujemy procedury o stosunkowo prostym do rozgryzienia algorytmie. Procedury optymalizowane pod kątem czasu realizacji lub miejsca zajmowanego w pamięci zawsze zawierają pewne haczyki, które ograniczają możliwości uniwersalnego stosowania. Zacznijmy więc od prostszych przykładów.

Przykłady procedur w języku ASSEMBLER 8080 (1)

W programach użytkowych, realizujących wyznaczone przez projektanta funkcje, często powtarzają się typowe procedury lub ciągi instrukcji. Należą do nich m.in. podprogramy realizujące niektóre operacje arytmetyczne, funkcje elementarne, operacje na tablicach, macierzach itp. Tego typu procedury zbierane są w biblioteki, fragmenty których użytkownik może dołączyć do swojego programu. Operacje mniej skomplikowane, z uwagi na szeroki ich zestaw nie są łączone w biblioteki, co oczywiście nie oznacza, że konstruktor powinien zawsze wymyślać je od początku. Każdy programista ma oczywiście własną bibliotekę prostych procedur, w których wykorzystuje typową dla siebie sekwencję instrukcji i sposób przekazywania parametrów. Dzięki stosowaniu wypróbowanych wzorców projektant może koncentrować swoją uwagę na optymalnym zapisaniu algorytmu, bez drobiazgowej analizy zawartości poszczególnych rejestrów podczas wykonywania typowych operacji.

Poniżej podajemy przykładowy zestaw prostych programów, realizujących stosunkowo często występujące operacje.

PROGRAMY ARYTMETYCZNE

W prostych systemach realizowanych na mikroprocesorze 8080 argumentami operacji arytmetycznych są zazwyczaj liczby 8- lub 16-bitowe. Mogą to być wartości odczytane z przetworników analogowo-cyfrowych, parametry nastaw, wartości zadane przez operatora itp. Argumenty przedstawiane są z reguły w kodzie binarnym lub BCD. W poniższych programach przyjęto, że argumenty znajdują się w rejestrach C i E (8 bitów) lub parach rejestrów BC i DE (16 bitów), natomiast wynik działania w rejestrze A (8 bitów) lub parze rejestrów HL (16 bitów).

Dodawanie

Dodawanie jest operacją najprostszą, z uwagi na to, że procesor 8080 ma w swoim zestawie rozkazów operacje dodawania liczb 8- i 16-bitowych. Oto zapis przykładowych procedur:

```
; DODAWANIE 8-BITOWYCH LICZB BINARNYCH
MOV     A, C        ; POBIERZ PIERWSZY SKŁADNIK
ADD     E           ; DODAJ DRUGI SKŁADNIK
```

Powyższa procedura zajmuje w pamięci 2 bajty i realizowana jest przez 9 cykli zegara (4,5 μs).

```
; DODAWANIE 16-BITOWYCH LICZB BINARNYCH
; ARGUMENTY W REJ. BC i DE; WYNIK W REJ. HL
MOV     L,E         ; POBIERZ PIERWSZY SKŁADNIK
MOV     H,D
DAD     B           ; DODAJ DRUGI SKŁADNIK
```

Powyższa procedura zajmuje w pamięci 3 bajty i realizowana jest przez 20 cykli zegara (10 μs).

```
; DODAWANIE 8-BITOWYCH LICZB BCD
; ARGUMENTY W REJ. C I E; WYNIK W REJ. A
MOV     A,C         ; POBIERZ PIERWSZY SKŁADNIK
ADD     E           ; DODAJ DRUGI SKŁADNIK
DAA                 ; KOREKCJA DZIESIĘTNA
```

Powyższa procedura zajmuje w pamięci 3 bajty i realizowana jest przez 13 cykli zegara (6,5 μs)

```
; DODAWANIE 16-BITOWYCH LICZB BCD
; ARGUMENTY W REJ. BC i DE; WYNIK W REJ. HL
MOV   A,C   ; POBIERZ LSB PIERWSZEGO SKŁADNIKA
ADD   E     ; DODAJ LSB DRUGIEGO SKŁADNIKA
DAA         ; KOREKCJA BCD LSB
MOV   L,A   ; ZAPAMIĘTAJ LSB SUMY
MOV   A,B   ; POBIERZ MSB PIERWSZEGO SKŁADNIKA
ADC   D     ; DODAJ MSB DRUGIEGO SKŁADNIKA BINARNIE
DAA         ; KOREKCJA BCD MSB
MOV   H,A   ; ZAPAMIĘTAJ MSB SUMY
```

Powyższa procedura zajmuje w pamięci 8 bitów i realizowana jest przez 36 cykli zegara (18 μs).

Odejmowanie

Procesor 8080 nie realizuje rozkazu odejmowania liczb 16-bitowych i dlatego tę operację należy wykonać jako dwukrotne odejmowanie liczb 8-bitowych. W przypadku liczb BCD należy znaleźć dopełnienie odjemnika do 100 i dodać do odjemnej.

```
; ODEJMOWANIE 8-BITOWYCH LICZB BINARNYCH
; ARGUMENTY W REJ. C I E; WYNIK W REJ. A
MOV     A,C         ; POBIERZ ODJEMNĄ
SUB     E           ; ODJEMNA W REJ. E, RÓŻNICA W REJ. A
```

Procedura zajmuje w pamięci 2 bajty i realizowana jest przez 9 cykli zegara (4,5 μs).

```
; ODEJMOWANIE 16-BITOWYCH LICZB BINARNYCH
; ARGUMENTY W REJ. BC I DE; WYNIK W REJ. HL
MOV     A,C         ; LSB ODJEMNEJ
SUB     E           ; ODEJMIJ LSB ODJEMNIKA
MOV     L,A         ; LSB RÓŻNICY
MOV     A,B         ; MSB ODJEMNEJ
SBB     D           ; ODEJMIJ MSB ODJEMNIKA Z POŻYCZKĄ
MOV     H,A         ; MSB RÓŻNICY
```

Procedura zajmuje w pamięci 6 bajtów i realizowana jest przez 38 cykli zegara (19 μs).

```
; ODEJMOWANIE 8-BITOWYCH LICZB BCD
; ARGUMENTY W REJ. C i E; WYNIK W AKUMULATORZE
MVI     A, 9AH      ; DOPEŁNIENIE DO 99 BEZ POŻYCZKI
SUB     E           ; DOPEŁNIENIE ODJEMNIKA
ADD     C           ; DODAJ ODJEMNĄ
DAA                 ; KOREKCJA DZIESIĘTNA WYNIKU
```

Procedura zajmuje w pamięci 5 bajtów i realizowana jest przez 15 cykli zegara (7,5 μs).

```
; ODEJMOWANIE 16-BITOWYCH LICZB BC
; ARGUMENTY W REJ. BC I DE ; WYNIK W REJ. HL
MVI     A, 9AH  ; DOPEŁNIENIE DO 100
SUB     E       ; DOPEŁNIENIE LSB ODJEMNIKA
ADD     C       ; DODAJ ODJEMNĄ
DAA             ; KOREKCJA DZIESIĘTNA LSB
MOV     L,A     ; ZAPAMIĘTAJ LSB WYNIKU
MVI     A, 99H  ; DOPEŁNIENIE DO 99
ACI     0       ; UWZGLĘDNIENIE POŻYCZKI
SUB     D       ; DOPEŁNIENIE MSB ODJEMNIKA
ADD     B       ; DODAJ MSB ODJEMNEJ
DAA             ; KOREKCJA DZIESIĘTNA MSB
MOV     H,A     ; ZAPAMIĘTAJ MSB WYNIKU
```

Procedura zajmuje w pamięci 14 bajtów i realizowana jest przez 55 cykli zegara (27 μs).

Mnożenie

Wszystkie operacje mnożenia, wykonywane przez procesor 8080 muszą być realizowane w sposób programowy. Mnożenie liczb całkowitych 8- i 16-bitowych bez znaku może być wykonane na rejestrach procesora, co znacznie skraca ich czas wykonania. W pierwszym przedstawionym przykładzie realizowane jest mnożenie liczb 8-bitowych (lub liczby 8-bitowej i 16-bitowej) dające wynik 16-bitowy (bez sprawdzania, czy wynik mnożenia mieści się w szesnastu bitach).

Drugi przykład przedstawia mnożenie dwóch liczb 16-bitowych, dające wynik 32-bitowy.

Trzeci przykład przedstawia mnożenie dwóch dwucyfrowych liczb BCD. Celem przyspieszenia operacji przedstawiono realizację algorytmu wykorzystującego tabliczkę mnożenia. Niestety takie rozwiązanie znacznie zwiększa zajmowany przez program obszar pamięci.

```
; MNOŻENIE 8-BITOWYCH LICZB BINARNYCH (LUB LICZBY
; 16-BITOWEJ PRZEZ 8-BITOWĄ)
; MNOŻNA W REJ. C (PARZE REJ. BC); MNOŻNIK W REJ. E
MNOZ:   MVI B, 0        ; ZEROWANIE DLA LICZBY 8-BITOWEJ
DMNOZ:  LXI H,0         ; ILOCZYN POCZĄTKOWY 0
        MVI D,8         ; LICZNIK BITÓW
        MOV A,E         ; MNOŻNIK W AKUMULATORZE
MNOZ1:  DAD H           ; PRZESUŃ WYNIK POŚREDNI
        RLC             ; NAJSTARSZY BIT MNOŻNIKA
        JNC MNOZ2
        DAD B           ; BIT MNOŻNIKA = 1, DODAJ MNOŻNĄ
MNOZ2:  DCR D           ; UWZGLĘDNIJ 8 BITÓW
        JNZ MNOZ1
        RET
```

Procedura zajmuje w pamięci 12 bajtów i realizowana jest przez ok. 75 μs.

```
; MNOŻENIE LICZB 16-BITOWYCH, WYNIK 32-BITOWY
; MNOŻNA W REJ. BC; MNOŻNIK W REJ. DE;
; ILOCZYN W REJ. DE (MSB) I HL (LSB)
MNOZ16: MVI     A, 10H ; LICZNIK BITÓW
        LXI     H,0    ; WARTOŚĆ POCZĄTKOWA
        PUSH    H      ; WYNIK NA STOS
MNBIT:  XCHG
        DAD     H      ; NAJSTARSZY BIT DO CY
        XCHG
        JNC     NDOD ; NIE DODAWAJ, JEŚLI CY = 0
        DAD     B      ; DO LSB DODAJ MNOŻNĄ
        JNC     NDOD ; JEST PRZENIESIENIE ?
        XTHL
        INX     H      ; UWZGLĘDNIJ PRZENIESIENIE W MSB
        XTHL
NDOD:   DCR     A      ; WSZYSTKIE BITY ?
        JNZ     MMSB
        POP     D      ; POBIERZ MSB
        RET
;
MMSB:   DAD     H      ; PRZESUŃ LSB W LEWO
        XTHL           ; POBIERZ MSB
        PUSH    PSW ; ZACHOWAJ PRZENIESIENIE Z LSB
        DAD     H      ; PRZESUŃ MSB
        POP     PSW
        JNC     MNCY ; BRAK PRZENIESIENIA Z LSB ?
        INX     H      ; DODAJ PRZENIESIENIE DO MSB
MNCY:   XTHL           ; MSB NA STOS
        JMP     MNBIT
```

Procedura zajmuje w pamięci 37 bajtów i realizowana jest przez ok. 2,2 ms.

```
; MNOŻENIE DWÓCH LICZB 2-CYFROWYCH BCD, Z WYKO-
; RZYSTANIEM TABLICZKI MNOŻENIA. WYNIK 4-CYFROWY.
; ARGUMENTY W REJ. 8 (WZ) I REJ. C (XY) ; WYNIK
; W REJ. HL
BCDMN:  LXI  H,0    ; WYNIK POŚREDNI
        CALL TMP    ; HL + W*X
        CALL TM10   ; 10*HL + X*Z
        CALL TM     ; HL + Y*W
        CALL TM10   ; 10*HL + Y*Z
        RET
;
TM10:   DAD  H      ; PRZESUŃ WYNIK POŚREDNI
        DAD  H      ; O JEDNĄ CYFRĘ
        DAD  H
        DAD  H
TMP:    MOV  A,B    ; PRZYGOTUJ MNIEJ ZNACZĄCY BAJT
        RLC         ; DO WYCIĘCIA
        RLC
        RLC
        RLC
        MOV  B,A
```

```
; PODPROGRAM MNOŻENIA Z TABLICZKĄ
TM:     MOV  A,B
        ANI  OFH    ; PRAWA CYFRA PIERWSZEGO CZYNNIKA
        MOV  E, A
        MOV  A,C
        ANI  0F0H   ; LEWA CYFRA DRUGIEGO CZYNNIKA
        ORA  E      ; POŁĄCZ CYFRY
        PUSH H      ; SCHOWAJ WYNIK CZĘŚCIOWY
        LXI  D,TABMN
        MOV  L,A    ; PRZYGOTUJ ADRES
        MVI  H,0    ; DLA TABLICZKI MNOŻENIA
        DAD  D
        MOV  A,M    ; ILOCZYN Z TABLICZKI
        POP  H
        ADD  L      ; DODAJ ILOCZYN CZĘŚCIOWY
        DAA         ; KOREKCJA BCD
        MOV  L,A
        MVI  A,0
        ADC  H      ; UWZGLĘDNIJ PRZENIESIENIE
        DAA         ; KOREKCJA BCD
        MOV  H,A    ; BARDZIEJ ZNACZĄCE CYFRY WYNIKU
        MOV  A,B    ; USTAW CYFRY
        MOV  B,C    ; DLA KOLEJNEGO
        MOV  C,A    ; WYCIĘCIA
        RET
```

```
; TABLICZKA MNOŻENIA
TABMN:  DB      00,00,00,00,00,00,00,00,00.....
        DB      00,01,02,03,04,05,06,07,08.....
        ......
        DB      00,09,18H,27H,36H,45H,54H.....
```

Procedura zajmuje w pamięci 56 bajtów, tabliczka mnożenia 144 bajty — razem 200 bajtów. Czas realizacji ok. 0,75 ms.

Dzielenie

Jako przykład operacji dzielenia przedstawione zostanie dzielenie dwóch liczb 16-bitowych. Wynikiem dzielenia jest 16-bitowy iloraz oraz 16-bitowa reszta.

```
; DZIELENIE 16-BITOWYCH LICZB BINARNYCH
; DZIELNA W REJ. BC ; DZIELNIK W REJ. DE,
; ILORAZ W REJ. BC, RESZTA W REJ. HL
DZIEL:  MOV   A,D
        CMA           ; NEGACJA REJESTRU D
        MOV   D,A
        MOV   A,E
        CMA           ; NEGACJA REJESTRU E
        MOV   E,A
        INX   D       ; UZUPEŁNIENIE DZIELNIKA W U2
        LXI   H,0     ; WARTOŚĆ POCZĄTKOWA RESZTY
        MVI   A,17    ; LICZNIK PĘTLI
DZIEL1: PUSH  H       ; ZAPAMIĘTAJ WARTOŚĆ RESZTY
        DAD   D       ; ODEJMIJ DZIELNIK
        JNC   DZIEL2  ; MIEŚCI SIĘ JEŻELI CY = 1
        XTHL          ; RÓŻNICA NA STOS
DZIEL2: POP   H       ; POBIERZ AKTUALNĄ RESZTĘ
        PUSH  PSW     ; ZAPAMIĘTAJ LICZNIK PĘTLI
        MOV   A,C
        RAL           ; PRZESUŃ DZIELNĄ W LEWO O JEDEN
                      BIT
        MOV   C,A     ; NA BIT NAJMNIEJ ZNACZĄCY
                      ; WPROWADŹ
        MOV   A,B     ; WARTOŚĆ ILORAZU
        RAL
        MOV   B,A
        MOV   A,L     ; PRZESUŃ RESZTĘ O JEDEN BIT
                      ; W LEWO
        RAL           ; NA BIT NAJMNIEJ ZNACZĄCY
                      ; WPROWADŹ
        MOV   L,A     ; NAJBARDZIEJ ZNACZĄCY BIT DZIELNEJ
        MOV   A,H
        RAL
        MOV   H,A
        POP   PSW     ; ODTWÓRZ LICZNIK PĘTLI
        DCR   A
        JNZ   DZIEL1
        DRA   A       ; KASUJ BIT CY
        MOV   A, H
        RAR           ; RESZTA PRZESUNIĘTA
        MOV   H,A     ; O JEDEN BIT ZA DALEKO
        MOV   A,L
        RER
        MOV   L,A
        RET
```

Procedura zajmuje w pamięci 44 bajty i jest realizowana przez ok. 1,3 ms.

W drugiej części artykułu przedstawione zostaną przykładowe procedury realizujące funkcje logiczne.

ADAM PLUTA
Przemysłowy Instytut Elektroniki
Warszawa

Nie chcemy, aby publikacje w mikroKLANIE adresowane do użytkowników konkretnych komputerów zostały zdominowane przez określony typ. Przeglądając dotychczasowe publikacje trudno oprzeć się wrażeniu lansowania ZX SPECTRUM. Nie wynika to jednak z preferencji redakcji. Jest to obecnie najpopularniejszy typ mikrokomputera w naszym kraju.

Ostatnio pojawił się nowy kandydat na gwiazdę — CPC 464. Poświęciliśmy mu już nieco miejsca, lecz mimo że polecamy go wielu potencjalnym użytkownikom, przygotowujemy również publikację, która odsłoni słabsze punkty tej konstrukcji (być może ukaże się ona już w usamodzielnionym mikroKLANIE).

Poniżej publikujemy krótki tekst praktyczny — dedykowany użytkownikom tego komputera. Ich liczba wzrasta, o czym może świadczyć zalążek powstającego w Warszawie klubu użytkowników AMSTRADA/SCHNEIDERA.

Zapamiętywanie obrazu w CPC 464

W mikrokomputerze AMSTRAD (SCHNEIDER) CPC 464 można zarejestrować w pamięci masowej obraz z ekranu wykorzystując w tym celu instrukcję SAVE. Instrukcja SAVE z opcją B dla tego typu zbioru (ang. Binary) pozwala zapisać na dysku lub kasecie zawartość obszaru pamięci w postaci binarnej. Parametrami instrukcji są m.in. adres początkowy i wielkość obszaru (jeżeli zapisywany byłby program, możliwe jest podanie jeszcze adresu startu). Pamięć obrazu zajmuje przestrzeń adresową o rozmiarach 16 K bajtów (4000H), znajdującą się w obszarze o adresach 0C000H... ...OFFFFH.

Instrukcja SAVE w postaci:

SAVE "!"+„FILENAME",B, &C000, & 4000

zapamiętuje na dysku lub kasecie aktualny obraz z ekranu, jako zbiór binarny. Wyświetlenie wcześniej zapamiętanego obrazu realizuje instrukcja LOAD w postaci:

LOAD "!"+„FILENAME"

Pierwszym elementem ciągu znaków podawanych w zapisie instrukcji jest wykrzyknik (zapis "!"+„FILENAME" jest równoważny z zapisem „! FILENAME"). Nie jest on jednak pierwszym znakiem faktycznej nazwy zbioru, gdyż jako znak specjalny jest zawsze usuwany. W przypadku pracy z pamięcią dyskową nie ma on żadnego znaczenia i w podanych instrukcjach można go pominąć.

W przypadku pracy z kasetami wykorzystanie znaku „!" pozwala uniknąć wykorzystywania ekranu do zapisu lub odczytu wszelkiego typu komunikatów i pytań dotyczących np.: znalezionych, ładowanych, zapisywanych zbiorów, numerów bloków i czynności operatorskich. Informacje tego typu zwykle są niepożądane w zapamiętanym lub odczytanym obrazie. Z tego samego powodu instrukcje zapisu i odczytu należy umieścić w programie (w trybie konwersacyjnym na obraz nałoży się zapis instrukcji).

Zapamiętywany obraz zajmuje nieco więcej niż 16 K bajtów, dlatego też należy upewnić się, czy dysponujemy wystarczająco dużym obszarem pamięci masowej.

Wygodniejszym urządzeniem do przechowywania informacji jest oczywiście dysk elastyczny, gdyż umożliwia szybki zapis i odczyt zbiorów. W przypadku pracy z kasetą szybkość zapisu można zwiększyć, przez wykonanie instrukcji SPEED WRITE 1. Przy odczycie ustalenie odpowiedniej szybkości następuje w sposób automatyczny (na podstawie bitów synchronizacji). Zwiększenie szybkości zapisu-odczytu wymaga stosowania kaset wyższej jakości.

Posługując się instrukcjami SAVE i LOAD w przedstawionej formie (z wykrzyknikiem) należy pamiętać o włączeniu REC i PLAY — w przypadku zapisu na kasecie — oraz PLAY — w przypadku odczytu — gdyż komunikaty żądające tych czynności operatorskich nie pojawią się na ekranie.

MAREK DORYWALSKI
Warszawa

μK

prowadzi:
Andrzej J. Piotrowski
tel. dom.: 48-22-85

dokończenie ze str. 12

CAPS SHIFT 6 (strzałka w dół) — przesuń kursor do następnego wiersza
CAPS SHIFT 7 (strzałka w górę) — przesuń kursor do poprzedniego wiersza
CAPS SHIFT 0 (DELETE) — usuń znak na lewo od kursora
EXTENDED MODE CAPS SHIFT 5 — przerzuć kursor na początek wiersza
EXTENDED MODE CAPS SHIFT 8 — przesuń kursor na koniec wiersza
EXTENDED MODE CAPS SHIFT 6 — przerzuć kursor na koniec ekranu
EXTENDED MODE CAPS SHIFT 7 — przerzuć kursor na początek ekranu
EXTENDED MODE B — przerzuć kursor na początek tekstu
EXTENDED MODE E — przerzuć kursor na koniec tekstu
EXTENDED MODE P — przerzuć kursor na poprzednią stronę
EXTENDED MODE N — przerzuć kursor na następną stronę
SYMBOL SHIFT S — wstrzymaj przewijanie; po naciśnięciu dolnego klawisza przewijanie tekstu zostanie wznowione
EXTENDED MODE Y — usuń część wiersza na prawo od kursora
EXTENDED MODE R — wstaw to, co zostało ostatnio usunięte przez EXTENDED MODE Y
EXTENDED MODE C — zakończ redagowanie
CAPS SHIFT BREAK SPACE — przerwij redagowanie, nie uwzględniając poprawek.

Poza trybem redagowania do poprawek można używać tych samych klawiszy co w edytorze, pod warunkiem, że pozostanie się w obrębie tego samego wiersza LOGO. EXTENDED MODE R powoduje napisanie wiersza ostatnio wprowadzonego przez ENTER, albo części wiersza skasowanej przez EXTENDED MODE Y, zależnie od tego, co zostało wykonane później.

BIBLIOGRAFIA

[1] Abelson H.: Logo for the Apple II. Byte Books, McGraw Hill 1982
[2] Abelson H.: A beginner's guide to Logo. Byte 7 no. 8 Aug. 1982, 88—112
[3] Abelson H., Klotz L. Jr.: Logo for the Apple II: Technical Manual. Terrapin Inc.
[4] Abelson H., Di Sessa A.: Turtle geometry: the computer as a medium for exploring mathematics. MIT Press 1981
[5] Aigle M.: Logo as a means of observation in a study of the learning process in: Lewis R., Tass D. E. eds.: Computers in Education. North Holland 1981, 796—797
[6] Allison L., Edmiston P.: A Logo survey. Australian Computer Bulletin. Oct. 1981, 44—43
[7] Bearden D.: 1, 2, 3, my computer and me. A Logo funbook for kids. Prentice Hall 1983
[8] Bearden D., Martin K., Muller J.: The turtle's sourcebook. Prentice Hall 1984
[9] Bitter G. G., Watson N. R.: Commodore 64 Logo primer. Reston 1985
[10] Bossuet G.: Introduire l'ordinateur a l'école? Presses Universitaires de France 1981
[11] Bossuet G., Fournier J., Leguyader C.: The Logo graphic „Aiguelongue" experience. in: Lewis R., Tass D. E. eds. Computers in Education. North Holland 1981, 795
[12] Brady J. M.: Introducing Logo education. Computer Education 19 1975, 24—27
[13] Conlan J., Inman D.: Sprites, a turtle, and TI Logo. Prentice Hall 1984
[14] Cugola M., Giaccaglini G.: Il mondo del Logo, un microcosmo in espansione. Bit 6 no. 37 Marzo 1983, 82—89
[15] Davidson L. J.: Apple Logo: Reference manual. Apple Computer Inc. 1982
[16] Du Boulay J.: Teaching mathematics through programming. Int. J. Math. Educ. Sci. Technol. 11 1982 no. 3, 347—360
[17] Feurzeig W., Lukas G.: Logo — a programming language for learning mathematics. Educational Technology March 1972, 33—46
[18] Gary G., Bitter G., Watson N. R.: Apple Logo primer. Reston 1983
[19] Goldberg A., Ross J.: Is the Smalltalk language for children? Byte 6 no. 8 Aug. 1981, 348—368
[20] Goldenberg E. P.: Special technology for special children. University Park Press 1979
[21] Goldenberg E. P.: Logo — a cultural glossary. Byte 7 no. 8 Aug. 1982, 210—228
[22] Goodyear P.: Commodore 64 Logo, a learning and teaching guide. Ellis Horwood 1984
[23] Goodyear P.: Logo, a guide to learning through programming. Ellis Horwood 1984
[24] Gravina R.: Smalltalk and the personal computer. Computer Education no. 35, 9—13
[25] Harvey B.: Why Logo? Byte 7 no. 8 Aug. 1982, 163—193
[26] Howe J., O'Shea T., Plane F.: Teaching mathematics through Logo programming: an evaluation study. in: Lewis R., Tagg E. D. eds. Computer Assisted Learning. North Holland 1980, 85—101
[27] Kay A., Goldberg A.: Personal dynamic media. Computer 10 1977 no. 3, 31—41
[28] Lawler R. W.: Designing computer-based microworlds. Byte 7 no. 8 Aug. 1982, 138—160
[29] Lothe H.: Logo Programmiersprache nicht nur für Kinder. Chip no. 1 Jan. 1983, 122—124
[30] McDougall A., Adams T.: Logo environments: the development of the language and its use in education and research. in: Sale A. H. J., Hawthorne G. eds. Proc. of the Ninth Australian Computer Conference, 1982, 116—132
[31] McDougall A., Adams T., Adams P.: Learning Logo on the Apple II. Prentice Hall 1982.
[32] Mellar H.: Logo learning. Personal Computer World Apr. 1982
[33] Nelson H.: Logo for personal computers. Byte 6 no. 6 June 1981, 36—44
[34] Nikolov R.: Logo, pervi klas. Sofia 1984
[35] Nikolov R., Sendova E.: Ezik i matematika. Logo, vtori klas. Sofia 1984
[36] O'Shea T.: Artificial Intelligence and Computer Based Education. Computer Education 1978 no. 30, 25—28
[37] Overall T. et al.: Learning with Logo at the Lamplighter School. Microcomputing Sept. 1981, 42—47
[38] Papert S.: Mindstorms: children, computers, and powerful ideas. Basic Books 1980
[39] Papert S.: Redefining childhood: The computer presence as an experiment in developmental psychology. in: Lavinston S. H. ed. Information Processing 80. North Holland 1980, 993—998
[40] Papert S.: Teaching children to be mathematician versus teaching about mathematics. Int. J. Math. Educ. Sci. Technol. 3 1977, 249—262, also in: Taylor R. P. ed.: The computer in the school, tutor, tool, tutee. Teachers College Press 1980, 177—196
[41] Parr M.: DIY Logo. Personal Computer World. Sep. 1982, 156—159
[42] Ross P., Howe J.: Teaching mathematics through programming: ten years on. in: Lewis R., Tagg D. E. eds.: Computers in Education. North Holland 1981, 193—148
[43] Ross P.: Introducing Logo for the Apple II computer, Texas Instruments 99/4A and Tandy Color Computer. Addison Wesley 1983
[44] Ross P.: Logo programming. Addison-Wesley 1983
[45] Solomon C., Harris D.: Language in the Logo computer culture. Proc. of the National Educational Computing Conference. U. Iowa 1979
[46] Solomon C.: Apple Logo: introduction to programming through turtle graphics. Logo Computer Systems Inc. 1982
[47] Solomon C.: Introducing Logo to children. Byte 7 no. 8 Aug. 1982, 196—208
[48] Sparer E.: Sinclair Logo 1. Turtle graphics. Sinclair Research Ltd 1984
[49] Sparer E.: Sinclair Logo 2. Programming reference manual. Sinclair Research Ltd 1984
[50] Tater D., Sobalvarro P.: The Terrapin Logo language tutorial. Terrapin Inc.
[51] Thornburg D. D.: Discovering Apple Logo. Addison-Wesley 1983
[52] Ury L.: A teknosbeka-grafika. Mikroszamitogep Magazin, no. 1, 1985, 16—19
[53] Waligórski S.: Język Logo i metody pracy z komputerem w szkole. I Krajowa Konferencja „Informatyka w szkole", Wałbrzych 1985.
[54] Waligórski S.: Wybrane języki i metody programowania w zastosowaniach. Współczesne kierunki rozwoju informatyki. Materiały Szkoły Jesiennej PTT Rydzyna 1984
[55] Watt D.: A comparison of the problem solving styles of two students learning Logo, a computer language for children. Creative Computing 5 1979 no. 12, 86—90
[56] Watt D.: Logo in the schools. Byte 7 no. 8 Aug. 1982, 116—134
[57] Weinreb W.: Problem solving with Logo. Byte 7 no. 11 Nov. 1982
[58] Weir S.: Logo and the exceptional child. Microcomputing Sept. 1981, 76—84
[59] Weir S., Watt D.: Logo: a computer environment for learning disabled students. The Computing Teacher 8 1981 no. 5, 11—19
[60] Williams G.: Logo for the Apple II, the TI-99/4A, and the TRS-80 Color Computer. Byte 7 no. 8 Aug. 1982, 230—290
[61] Wills S.: Computer education in Tasmanian schools. The Australian Computer Bulletin. Aug. 1980, 22—28
[62] Wills S.: Turtles for teaching computing. in: Lewis R., Tagg D. E. eds. Computers in Education. Nort Holland 1981, 795—796.

Zamknięcie układu mikroprocesora w układ VLSI uniemożliwia śledzenie jego funkcjonowania. Możemy jedynie na podstawie opisu architektury (w postaci rysunku) analizować jego działanie. Z kolei, dla celów dydaktycznych konieczne jest stworzenie studentom możliwości zaprojektowania i krokowego analizowania realizacji typowych rozkazów procesorów. Komputer EW jest przykładem rozwiązania problemu wspomagania nauczania informatyki za pomocą stosunkowo prostego rozwiązania sprzętowego. Wielu Czytelnikom konstrukcja ta pomoże przypomnieć sobie, sposób funkcjonowania nowoczesnego procesora. (Red.).

ADAM GACEK
STANISŁAW KOZIELSKI
PIOTR STRÓŻYNA
Politechnika Śląska
Gliwice

EW — elastycznie mikroprogramowany komputer dydaktyczny

Prezentowany poniżej prosty komputer o symbolu EW został skonstruowany na potrzeby procesu dydaktycznego. Komputer ten stanowiący rozwinięcie koncepcji komputera o nazwie W, którego budowa została przedstawiona w pracach [3, 5, 6, 7] służy do analizy przesłań międzyrejestrowych i organizacji cyklu rozkazowego oraz modelowania listy rozkazów. Struktura komputera W obejmuje jedynie procesor i pamięć operacyjną. Dysponuje on prostą listą rozkazów, realizowaną przez układ sterujący, pracujący w dwóch wariantach: sterowania sztywnego i mikroprogramowanego.

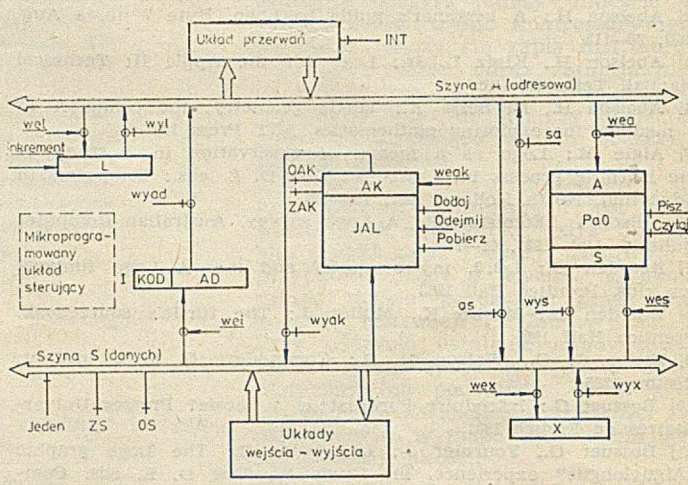
Wersja komputera EW, oprócz procesora ma dodatkowo układ obsługi przerwań oraz układ systemu wejścia-wyjścia. Celem rozszerzenia zakresu ćwiczeń dydaktycznych konstrukcję układu sterującego tego komputera oparto na bloku pamięci mikroprogramowanej z możliwością elastycznego programowania przez wprowadzanie mikroprogramów z pulpitu operatora. Pozwala to realizować — w ramach danej struktury maszyny — większy zbiór rozkazów, aniżeli wynikałoby to z ograniczonej do ośmiu bitów długości słowa rozkazowego (a w szczególności 3-bitowego kodu rozkazu). Ten sam zbiór kodów rozkazów wykorzystywany jest wtedy dla różnych grup rozkazowych. Dla każdej z tych grup pamięć wypełniana jest odrębnymi mikroprogramami, realizującymi funkcje rozkazów danej grupy. Umożliwia to zastosowanie znacznie mniejszej pamięci mikroprogramów, niż wynikałoby to z potrzeb jednorazowej realizacji pełnej listy rozkazów.

Koncepcja komputera EW odpowiada wymaganiom związanym z organizacją zajęć dydaktycznych. Podział zajęć na jednostki dydaktyczne narzuca bowiem konieczność oddzielnej analizy poszczególnych bloków funkcjonalnych, a więc również oddzielnego rozpatrywania odpowiednich grup rozkazów dotyczących tych bloków.

Elastyczność mikroprogramowania pozwala na łatwe porównywanie różnych realizacji podobnych rozkazów oraz wprowadzanie elementów optymalizacji do procesu ustalania listy rozkazów.

ARCHITEKTURA KOMPUTERA

Architektura jednostki centralnej komputera EW przedstawiona została na rys. 1. Przesłania międzyrejestrowe są kontrolowane przez sygnały mikrosterujące. Sygnały te sterują również pracą jednostki arytmetyczno-logicznej, pamięci operacyjnej oraz pozostałych bloków funkcjonalnych maszyny. Na rys. 1 zaznaczono ważniejsze sygnały mikrosterujące, rozróżniając sygnały impulsowe (podkreślone) i statyczne.



Rys. 1. Architektura jednostki centralnej komputera EW

Szyna danych ma osiem bitów (jest to długość słowa maszynowego). Szyna adresowa jest 5-bitowa, co pozwala na bezpośrednie adresowanie 32 komórek pamięci operacyjnej.

Procesor komputera EW składa się z:

- rejestru rozkazów (I), z wyodrębnionymi podrejestrami kodu operacji (KOD) i argumentu rozkazu (AD)
- licznika rozkazów (L)
- jednostki arytmetyczno-logicznej (JAL)
- rejestru akumulatora (AK)
- rejestru ogólnego przeznaczenia (X).

Ponadto w procesorze występują:

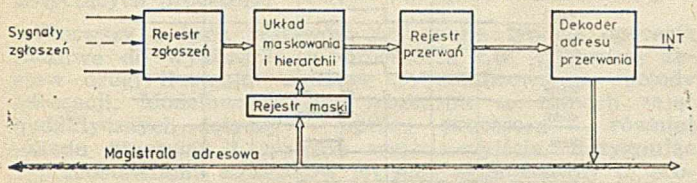
- połączenia między szynami: danych i adresową, w obu kierunkach
- układy wyznaczające następujące sygnały stanu maszyny:

— 0AK (zerowa zawartość akumulatora)
— ZAK (ujemna zawartość akumulatora)
— 0S (zero na szynie S)
— ZS (ujemna liczba na szynie S)
— INT (żądanie przerwania)
— ORO (zgłoszenie operatora z klawiatury sterującej).
Sygnały stanu mogą być testowane w układzie sterowania maszyny. Warto również zwrócić uwagę na istnienie sygnału mikrosterującego „jeden", ustawiającego liczbę 1 na magistrali danych.

Struktura procesora maszyny pozwala na realizację bardziej złożonych rozkazów, będących ilustracją możliwości funkcjonalnych współczesnych komputerów.

Układ przerwań komputera charakteryzuje się następującymi cechami (rys. 2):

- zgłoszenia są generowane na 4 poziomach
- źródłami zgłoszeń są przyciski klawiatury sterującej oraz system wejścia-wyjścia maszyny
- istnieje możliwość maskowania poszczególnych poziomów
- adres przerwania jest generowany automatycznie (układ wektorowy).

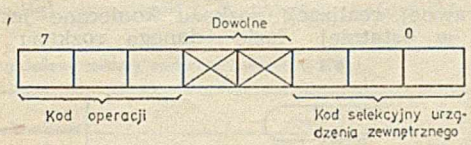


Rys. 2. Uproszczony schemat układu przerwań

Zgłoszenie przerwań (z klawiatury funkcyjnej oraz systemu wejścia-wyjścia) są przyjmowane do rejestru zgłoszeń. Wyjścia z tego rejestru mogą być maskowane zależnie od zawartości rejestru maski. Przy zgłoszeniach na kilku poziomach, wybór poziomu o najwyższym priorytecie następuje w układzie hierarchii. Zgłoszenie wybrane powoduje generację sygnału przerwania INT, będącego sygnałem stanu wykorzystanym w układzie sterowania.

System we-wy komputera obejmuje jedynie dwa układy sprzęgów: czytnika oraz perforatora (dziurkarki) taśmy papierowej. Obsługa urządzeń zewnętrznych może odbywać się zarówno z wykorzystaniem układu przerwań, jak i bez jego pośrednictwa.

Należy zaznaczyć, że wszystkie sygnały mikrosterujące dla układów we-wy są bramkowane sygnałem kodu selekcyjnego. Sygnały sterujące są wysyłane do wszystkich układów sprzęgów, jednak efekt wywołują one tylko w układzie wybranym przez kod selekcyjny, określony przez trzy najmniej znaczące bity rejestru rozkazów. Fakt ten narzuca format rozkazów wejścia-wyjścia przedstawiony na rys. 3.



Rys. 3. Format rozkazów wejścia-wyjścia

Zestaw dwóch urządzeń zewnętrznych o przeciwnych kierunkach transmisji jest wystarczający do zobrazowania pracy systemu we-wy dla celów dydaktycznych. System ten można w łatwy sposób rozbudować, dołączając nowe sprzęgi współpracujące z procesorem na zasadach analogicznych, jak układy sprzęgów czytnika i perforatora.

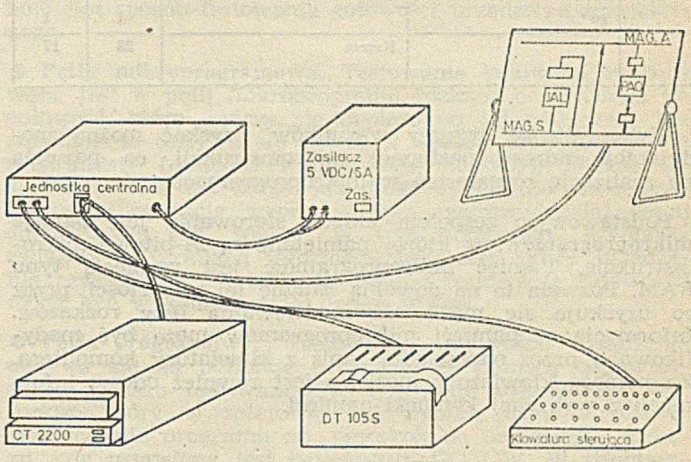
Tablica synoptyczna pozwala bieżąco kontrolować zawartości rejestrów oraz sygnałów mikrosterujących. Jest to nieodzowne dla celów dydaktycznych, gdyż pozwala na krokowe śledzenie realizacji rozkazów.

Funkcję indykatorów stanów rejestrów i sygnałów mikrosterujących pełnią wyświetlacze na diodach elektroluminescencyjnych.

Klawiatura sterująca w połączeniu z tablicą synoptyczną pełni funkcje pulpitu operatora maszyny. Umożliwia ona sterowanie pracą komputera, a w szczególności:

- wybór trybu pracy maszyny (praca krokowa — mikroinstrukcja po mikroinstrukcji); praca krokowa — (rozkaz po rozkazie); praca cykliczna
- inicjowanie i zatrzymywanie pracy
- zerowanie ogólne układów
- generowanie zgłoszeń na dwóch poziomach układu przerwań
- wpisywanie informacji do pamięci operacyjnej oraz do pamięci mikroprogramów.

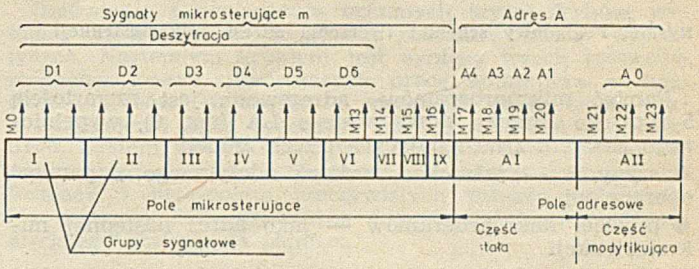
Schematycznie wygląd całego zestawu laboratoryjnego komputera przedstawiono na rys. 4.



Rys. 4. Konfiguracja zestawu laboratoryjnego komputera EW

Najciekawszym z dydaktycznego punktu widzenia elementem komputera EW jest mikroprogramowany układ sterowania, dlatego zostanie on opisany bardziej szczegółowo niż pozostałe bloki funkcjonalne maszyny.

Mikroinstrukcje stanowią 24-bitowe słowa sterujące realizacją rozkazów w komputerze. Słowo mikroinstrukcji podzielone jest na dwa pola: mikrosterujące i adresowe (rys. 5).



Rys. 5. Format mikroinstrukcji

Pozycje pola mikrosterującego o numerach od M0 do M13 są deszyfrowane, dając 25 sygnałów mikrosterujących. Sygnały odpowiadające pozycjom M14, M15, M16 są wykorzystywane bezpośrednio, jako sygnały mikrosterujące. Łącznie więc, w układzie sterowania komputera wytwarzanych jest 28 sygnałów mikrosterujących. Funkcje poszczególnych grup sygnałowych oraz symboliczny opis przyporządkowanych im sygnałów mikrosterujących przedstawia tabela.

Najmniej znaczące pozycje słowa mikroinstrukcji M17—M23 są wykorzystywane do formowania adresu następnej mikroinstrukcji.

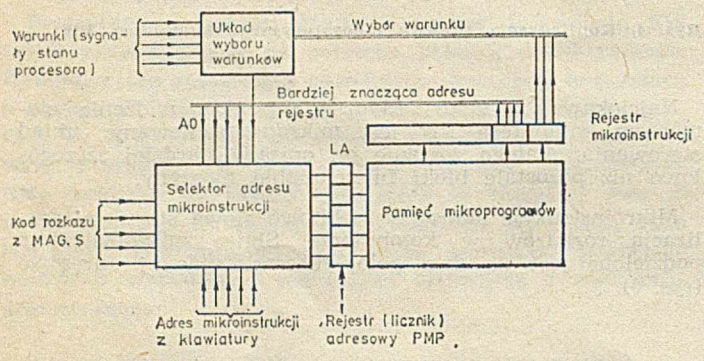
Pole adresowe mikroinstrukcji jest podzielone na dwie części, z których pierwsza (M17, M18, M19, M20) tworzy bezpośrednio cztery bardziej znaczące bity adresowe A4, A3, A2, A1, natomiast druga część pozwala na wypracowanie najmniej znaczącego bitu adresowego (A0), w zależności od wybranego sygnału stanu komputera.

Wykaz sygnałów mikrosterujących

Grupa sygnałów	Funkcja grupy sygnałów	Symbole sygnałów	Liczba sygnałów	Liczba bitów kodujących
I	Wyjście na magistralę S	wyg, rwe, wyx, jeden, wyak, as, wys	7	3
II	Wyjście z magistrali S	wej, wes, rwy, weak, wex	5	3
III	Wyjście na magistralę A	wyad, wyl, wyap, sa	4	2
IV	Wyjście z magistrali A	wel, wem, wea	3	2
V	Funkcje JAL	dodaj, odejmij, pobierz	3	2
VI	Inne sygnały sterujące	stop, inkrement, start	3	2
VII	Zerowanie przerwania	rint	1	1
VIII	Przyjęcie zgłoszenia	eni	1	1
IX	Koniec cyslu rozkazowego	kcm	1	1
		Razem	28	17

Wykorzystując sygnały warunków, uzyskać można modyfikację adresu następnej mikroinstrukcji, co pozwala na realizację rozgałęzień w mikroprogramach.

Podstawowym zespołem układu sterowania jest **pamięć mikroprogramów**, w której pamiętane są 24-bitowe mikroinstrukcje. Pamięć mikroprogramów jest pamięcią typu RAM. Pozwala to na dowolną zmianę jej zawartości, przez co uzyskuje się możliwość modelowania listy rozkazów. Informacja w pamięci mikroprogramów może być modyfikowana przez odpowiedni zapis z klawiatury komputera. Za pomocą klawiatury możliwy jest również odczyt informacji z wybranej komórki pamięci.



Rys. 6. Poglądowy schemat tworzenia adresu mikroinstrukcji

Pamięć mikroprogramów adresowana jest zawartością 5-bitowego licznika adresowanego LA (rys. 6), wypełnionego adresem z następujących trzech źródeł:

- szyny S — jako kod rozkazu odczytanego z pamięci operacyjnej
- pamięci mikroprogramów — jako adres następnej mikroinstrukcji
- klawiatury komputera — jako adres komórki pamięci mikroprogramów, wybranej do kontroli lub modyfikacji.

Przepływ informacji stanowiącej adres dla pamięci mikroprogramów z pożądanego źródła jest realizowany przez odpowiednie wysterowanie przełącznic adresowych.

ORGANIZACJA CYKLU ROZKAZOWEGO

Cykl rozkazowy komputera EW obejmuje następujące etapy realizacji:

- odczyt rozkazu
- inkrementację licznika rozkazów
- wykonanie funkcji rozkazu
- rozpoznanie i obsługę przerwania.

Zasady organizacji cyklu rozkazowego zilustrowane zostaną na przykładzie realizacji rozkazu DOX o funkcji określonej zapisem:

(X) + ((AD)) → AK

Treścią rozkazu jest dodanie zawartości rejestru X i komórki pamięci wskazanej w części adresowej rozkazu.

Na pełny cykl tego rozkazu składają się następujące operacje elementarne:

(L)→A: przygotowanie adresu rozkazu

((A))→I: odczyt rokazu z pamięci operacyjnej i przesłanie go do rejestru rozkazów

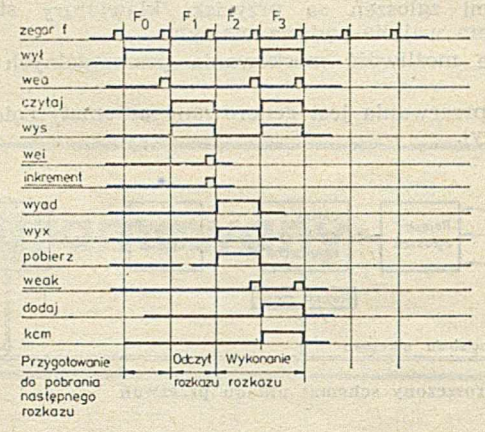
(L)+1→L: inkrementacja licznika rozkazów

(X)→AK: przesłanie pierwszego argumentu do akumulatora

(AD)→A: przygotowanie adresu drugiego argumentu

(AK)+((A))→AK: odczyt drugiego argumentu i dodanie go do zawartości akumulatora.

Wymienione operacje elementarne nie obejmują etapu rozpoznania i obsługi przerwania. Analiza tego etapu przedstawiona zostanie w dalszej części artykułu.



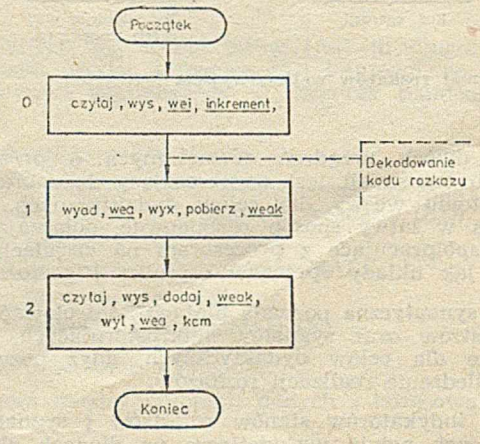
Rys. 7. Przebiegi sygnałów mikrosterujących w przykładowym cyklu rozkazowym

Przebiegi czasowe sygnałów mikrosterujących przedstawiono na rys. 7. Przyjęto, że wszystkie sygnały mikrosterujące są funkcjami taktu zegarowego.

Impulsy f służą do formowania impulsowych sygnałów wpisu do rejestrów, natomiast cykl zegara F (nazywany też fazą) jest podstawą syntezy statycznych sygnałów otwierających wyjścia na szynę rejestrów, które sterują pracą jednostki arytmetyczno-logicznej oraz pamięci operacyjnej.

Dla ułatwienia analizy, kolejne cykle zegarowe (fazy) zostały ponumerowane. Zauważamy, że w fazie F_0 następuje przesłanie zawartości licznika rozkazów do rejestru adresowego pamięci operacyjnej. Jest to przygotowanie do pobrania rozkazu i operację tę można wykonać w ostatniej fazie poprzednio wykonywanego rozkazu. Zgodnie z tą zasadą w fazie F_3 umieszczono sygnały „wyl" i „wea" realizujące przygotowanie do pobrania następnego rozkazu. Przy takim rozwiązaniu można zrezygnować z wyodrębniania fazy F_0.

Do poprawnej realizacji rozkazu konieczne jest wygenerowanie w ostatniej fazie danego rozkazu (tzn. F_3



Rys. 8. Schemat blokowy przykładowego mikroprogramu

z rys. 7) sygnału „kcm", umożliwiającego w fazie następnej (tzn. F_1) wybór właściwego mikroprogramu wykonania rozkazu. Powyższe uwagi pozwalają dokonać syntezy mikroprogramu dla analizowanego rozkazu. Mikroprogram ten przedstawiony został w postaci schematu blokowego na rys. 8. Każdy blok tego schematu odpowiada jednej mikroinstrukcji.

Mikroinstrukcja o numerze 0 wykonywana jest dla wszystkich rozkazów, a adres następnej mikroinstrukcji jest ustalany na podstawie kodu rozkazu. Przy pracy ciągłej mikroprogram każdego rozkazu kończy się przejściem do mikroinstrukcji o numerze 0.

MODELOWANIE LISTY ROZKAZÓW

Struktura komputera EW pozwala realizować większość typowych rozkazów komputerowych. 8-bitowa długość słowa rozkazowego ogranicza jednak istotnie liczbę rozkazów, którym można przyporządkować różne kody. Wykorzystując łatwość programowania pamięci mikroprogramów, można jednak zmieniać zestaw rozkazów, wybierając do prezentacji tylko określoną część możliwości komputera.

Ten sam zestaw kodów będzie więc każdorazowo odpowiadał innej grupie rozkazów, zależnie od aktualnej zawartości pamięci mikroprogramów. W tabeli 2 przedstawiono przykładowo dwa alternatywne zestawy rozkazów dotyczących procesora.

Pierwszy z tych zestawów prezentuje typowe operacje możliwe do wykonania w procesorze EW, natomiast zestaw drugi ilustruje możliwe do zrealizowania metody adresacji. Modelowanie listy rozkazów w ramach zajęć dydaktycznych dotyczy — oprócz procesora — również układu przerwań i systemu wejścia-wyjścia. Rezygnując z przedstawiania szczegółów [1] scharakteryzujemy tę problematykę w sposób ogólny.

Analiza pracy układu przerwań w kontekście modelowania listy rozkazów wymaga określenia:

- zakresu obsługi przerwania, realizowanej automatycznie w cyklu rozkazowym
- czynności, związanych z powrotem do przerwanego programu.

Obsługa przerwania w cyklu rozkazowym wymaga zapamiętania stanu procesora w momencie przerwania i przejścia do programu obsługi przerwania. Stan procesora (po zakończeniu realizacji kolejnego rozkazu) jest określony zawartością licznika rozkazów **L**, rejestru akumulatora **AK** oraz rejestru roboczego **X**. W cyklu rozkazowym rozważane są trzy różne warianty obsługi przerwania:

- przejście do programu obsługi przerwania następuje bez zapamiętywania rejestrów procesora
- w cyklu obsługi przerwania jest zapamiętywana zawartość licznika rozkazów, natomiast zawartości pozostałych rejestrów są zapamiętywane na początku programu obsługi przerwania
- w cyklu obsługi przerwania zapamiętywane są automatycznie zawartości wszystkich rejestrów określających stan procesora.

Zauważmy, że kolejne warianty pozwalają uprościć program obsługi przerwania, wymagają natomiast większego czasu na realizację, a także większej liczby zajmowanych komórek pamięci (czasami niepotrzebnie).

Stosownie do przedstawionych wariantów obsługi przerwania, rozważane są różne sposoby wyjścia z programu obsługi przerwania. Powrót z programu obsługi przerwania wymaga wyzerowania układu przerwań i odtworzenia stanu procesora. W różnych wariantach rozkazu powrotu odtwarzana jest zawartość tych rejestrów, które były zapamiętywane automatycznie w trakcie obsługi przerwania w cyklu rozkazowym.

Biorąc pod uwagę możliwości sprzętowe komputera EW można wyróżnić trzy różne — z programowego punktu widzenia — sposoby realizacji wymiany informacji, które określimy jako: pętla mikroprogramowa (połączenie mikroprogramowe), pętla programowa (połączenie programowe), praca z przerwaniami. Kryterium wyodrębnienia tych trybów jest sposób testowania gotowości urządzenia zewnętrznego.

- **Pętla mikroprogramowa.** Testowanie gotowości realizowane jest w pętli mikroprogramu rozkazu oczekiwania na gotowość, który kończy się dopiero po jej nadejściu. Testowanie to odbywa się zatem w sposób niewidoczny dla programisty (poziomu wynikowego maszyny). Rozkaz testowania trwa tak długo, jak długo urządzenie zewnętrzne wykonuje operację wyprowadzania (wprowadzania) danej elementarnej (np. znaku). W tym przypadku czynności związane z wymianą informacji (inicjacja pracy urządzenia zewnętrznego, przesłanie danej i testowanie gotowości) mogą być wykonane przez jeden rozkaz.

- **Pętla programowa.** Testowanie gotowości urządzenia odbywa się na poziomie programu za pomocą odpowiedniego rozkazu testowania. W tym celu wykonywany jest rozkaz, który w zależności od wyniku testu umożliwia rozgałęzienie programu np. przeskok do następnego rozkazu, gdy ustawiony jest przerzutnik GOTOW. W rozwiązaniu tym poszczególne operacje we-wy są realizowane przez odrębne rozkazy. Argumentem wszystkich rozkazów we-wy jest kod selekcyjny urządzenia zewnętrznego.

- **Wykorzystanie przerwań.** Rozkazy we-wy powodują jedynie inicjowanie pracy urządzeń peryferyjnych oraz przesyłanie danych (między procesorem a układami sprzęgu). Testowanie gotowości osiąga się przez przerwanie. Nie jest potrzebny rozkaz testowania gotowości. W ogólnym przypadku, po wstępnej inicjacji pracy urządzenia, można przejść do wykonywania procedury nie związanej z wymianą informacji. Procedura ta będzie przerywana za każdym razem po zgłoszeniu gotowości przez czytnik lub perforator.

Omówienie i porównanie opisanych wyżej trybów wymiany informacji z otoczeniem ma duże walory dydaktyczne. Następnym krokiem jest synteza trzech zestawów rozkazów we-wy, ilustrujących pracę komputera w każdym z trybów. W oparciu o stworzone przez siebie rozkazy, studenci piszą procedury obsługi urządzeń zewnętrznych. System we-wy komputera EW, mimo że jest bardzo prosty, pozwala dobrze ilustrować problemy wymiany informacji z otoczeniem rzeczywistych maszyn cyfrowych. Na zestawie EW nie można jedynie zrealizować bezpośredniego dostępu do pamięci.

Tabela 2. Przykładowe zestawy rozkazów komputera EW

Lp.	Kod rozkazu	Zestaw I		Zestaw II	
		nazwa rozkazu	funkcja rozkazu	nazwa rozkazu	funkcja rozkazu
1	000	POBIERZ DO AK	((AD)→ AK	SKOK BEZWARUNKOWY POŚREDNI	((AD)) → L
2	001	ŁADUJ AK	(AK) → (AD)	SKOK WARUNKOWY BEZPOŚREDNI	((AK) = 0) ⇒ (AD) → L
3	010	DODAJ	(AK) + ((AD)) → AK	POBIERZ DO AK INDEKSOWO	((AD + (X)) → AK
4	011	ODEJMIJ	(AK) — ((AD)) → AK	POBIERZ DO AK POŚREDNIO	(((AD))) → AK
5	100	SKOK BEZWARUNKOWY	(AD) → L	POBIERZ DO AK INDEKSOWO POŚREDNIO	(((AD)) + (X)) → AK
6	101	SKOK PRZY MINUSIE	((AK) < 0) ⇒ ⇒ (AD) → L	SKOK WZGLĘDNY	(L) + (AD) → L
7	110	POBIERZ X	(AK) → X	POBIERZ DO X BEZPOŚREDNIO	((AD)) → X
8	111	PRZEŚLIJ X	(X) → AK	POBIERZ DO X NATYCHMIASTOWO	(AD) → X

AD oznacza część rejestru rozkazów zawierających adres argumentu

Przedstawiony w artykule komputer EW oraz wspomniany we wstępie komputer W są wykorzystywane w Politechnice Śląskiej, w ramach ćwiczeń laboratoryjnych z przedmiotu Podstawy Informatyki, prowadzonego przez prof. dr. inż. Stefana WĘGRZYNA, który zainicjował prace konstrukcyjne nad tymi modelami.

Komputer W dysponuje gotową listą rozkazów i umożliwia tylko śledzenie ich realizacji na poziomie przesyłów międzyrejestrowych.

Własności komputera EW umożliwiają:

- analizę współpracy układu przerwań i układu wejścia-wyjścia z procesorem
- modelowanie listy rozkazów dla zrealizowania założonych funkcji rozkazów
- optymalizację listy rozkazów, dla uzyskania pożądanego podziału funkcji, realizowanych w komputerze w sposób układowy i programowy.

Obecnie wykorzystywanych jest sześć modeli obu typów komputerów. Zebrane w początkowym okresie eksploatacji spostrzeżenia doprowadziły do jednej bardziej istotnej modyfikacji układowej, a mianowicie alternatywnej realizacji pamięci mikroprogramów w postaci pamięci przełącznikowej. Pamięć ta stanowi matrycę przycisków stabilizowanych, których stan reprezentują poszczególne bity mikroprogramu. Ułatwia to kontrolę tworzonych mikroprogramów i przyspiesza proces ich uruchamiania. Dotychczasowe doświadczenia związane z eksploatacją maszyny potwierdziły użyteczność przedstawionej koncepcji komputera EW i jego możliwości funkcjonalnych jako specjalizowanej maszyny cyfrowej do zastosowań dydaktycznych.

LITERATURA

[1] Gacek A., Kozielski S., Stróżyna P.: Modelowanie listy rozkazów dla potrzeb komunikacji komputera dydaktycznego z otoczeniem. Zeszyty Naukowe Politechniki Śląskiej, Informatyka, zeszyt 5, 1984

[2] Gacek A., Kozielski S., Stróżyna P.: Organizacja komputera dydaktycznego z możliwością modelowania listy rozkazów. Zeszyty Naukowe Politechniki Śląskiej, zeszyt 5, 1984

[3] Konsek M.: Organizacja cyklu rozkazowego jako regulamin obsługi zgłoszeń. Podstawy Sterowania, t. 6, z. 1, 1976

[4] Stróżyna P.: Środki wspomagające mikroprogramowanie dydaktycznego komputera EW. Zeszyty Naukowe Politechniki Śląskiej, Informatyka, zeszyt 5, 1984

[5] Węgrzyn S.: Podstawy informatyki. PWN, Warszawa, 1982

[6] Węgrzyn S., Konsek M.: Przesyły międzyrejestrowe w programowanych maszynach cyfrowych o adresowanych rejestrach pamięci. Podstawy Sterowania, t. 5, z. 4, 1975

[7] Węgrzyn S., Konsek M., Mrózek A.: Mikrosterowanie i mikroprogramowanie. Podstawy Sterowania, t. 6, z. 1, 1976.

Samotesty

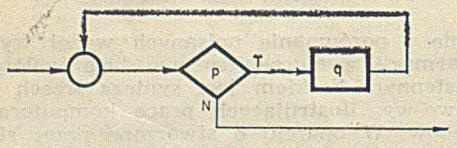
IV/A. Metodyka programowania

Samotesty, opracowane i opublikowane przez Association for Computing Machinery (ACM), nie stanowią standardu egzaminacyjnego i są przeznaczone wyłącznie do samodzielnego sprawdzania własnej wiedzy.

Narzędzia i metody konstruowania programów

PYTANIA

1. Rozważ następujący fragment schematu blokowego (ang. flowchart):



Która z przytoczonych poniżej konstrukcji programowania strukturalnego (ang. structured programming) ściśle odpowiada temu fragmentowi?;

a) WHILE p do q
b) DO q UNITIL p
c) IF p THEN q
d) WHILE q THEN p

2. Dana jest przykładowa tablica decyzyjna (ang. decision table):

Przykład tablicy decyzyjnej

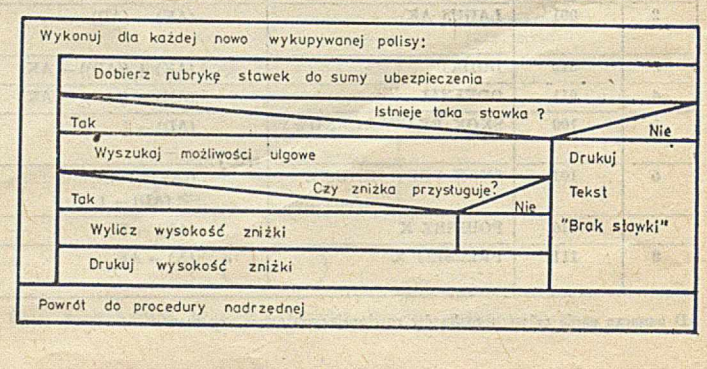
Ostatni rekord transakcyjny	nie	nie	nie	tak	nie	tak
Koniec pliku głównego	nie	nie	nie	nie	tak	tak
Porównaj klucze rekordów pliku głównego (g) i transakcyjnego (t)	g < t	g = t	g > t			
Drukuj klucz transakcji wraz ze wskazaniem rodzaju błędu			x		x	
Aktualizuj treścią transakcji rekord główny oraz dane sumaryczne		x				
Zapisz nowy rekord główny	x			x		
Czytaj rekord transakcyjny		x	x		x	
Czytaj kolejny rekord główny	x			x		
Drukuj dane sumaryczne oraz zakończ przetwarzanie						x
Powtórz przetwarzanie według tablicy	x	x	x	x	x	x

Któremu układowi warunków, jakie zaznaczono w tej tablicy, odpowiada wykonanie następującego zestawu czynności: „zapisz nowy rekord główny, odczytaj kolejny rekord główny i powtórz przetwarzanie według niniejszej tablicy"?

a) brak wystąpienia ostatniego rekordu transakcyjnego; brak wystąpienia końca pliku głównego; klucz rekordu głównego zgodny z kluczem rekordu transakcyjnego
b) brak wystąpienia ostatniego rekordu transakcyjnego; brak wystąpienia końca pliku głównego; klucz rekordu głównego mniejszy od klucza rekordu transakcyjnego
c) brak wystąpienia ostatniego rekordu transakcyjnego; wystąpienie końca pliku głównego
d) wystąpienie ostatniego rekordu transakcyjnego; brak wystąpienia końca pliku głównego.

3. Jakiego typowego procesu przetwarzania danych dotyczy tablica decyzyjna podana w poprzednim pytaniu?
   a) sortowania pliku (ang. file sorting)
   b) aktualizacji pliku (ang. file updating)
   c) scalania pliku (ang. file merging)
   d) zakładania pliku (ang. file creation).

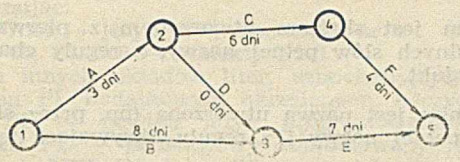
4. Rozważ przykładowy schemat Nassiego-Schneidermana:



Które z poniższych stwierdzeń jest (są) b ł ę d n e w stosunku do treści logicznej tego schematu?

a) pętla iteracyjna jest wyraźnie uwidoczniona i poprawnie zdefiniowana
b) wszystkie warunki typu IF...THEN...ELSE są wyraźnie uwidocznione i poprawnie zdefiniowane
c) występują pewne nieoznaczoności strukturalne i dowolności wykonywania poszczególnych dróg logicznych
d) niewypełniona rubryka (poniżej drugiego NIE) najprawdopodobniej nie jest błędem.

5. Rozpatrz powiązania czasowe w następującej sieci typu PERT:



Która z wymienionych poniżej sekwencji odpowiada ścieżce krytycznej?

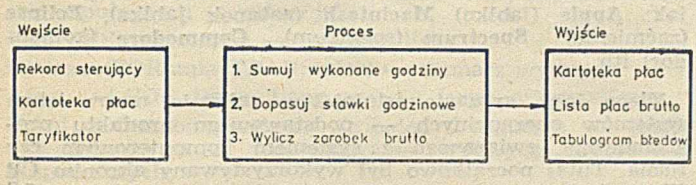
a) A-C-F b) A-D-E c) B-E-F d) B-E

6. Które z podanych niżej określeń najlepiej określa cel wyszukiwania błędów programowych (ang. debugging)?

a) usunięcie z programu wykrytych błędów
b) wyznaczenie stopnia wykrywalności błędów programowych
c) uzyskanie prawidłowego oprogramowania
d) dokumentowanie dużego programu.

7. Wymień przynajmniej cztery różne sposoby ujawniania błędów (ang. debugging) stosowane na etapie tworzenia oprogramowania (ang. program development).

8. Rozważ następujący schemat:



Czy można określić, abstrahując od jego sensu algorytmicznego, którego z poniżej wymienionych haseł najprawdopodobniej dotyczy ten schemat?

a) TOP-DOWN b) STRUCTURED
c) HIPO d) FIFO

ROZWIĄZANIA

1 a, 2 b, 3 b, 4 c, 5 d, 6 b, 7 — patrz komentarze, 8 c.

LITERATURA I KOMENTARZE

ad 1. Aron J.: The Program Development Process. Part. I. Addison-Wesley, pp. 137—139

ad 2. Cushing B.: Accounting Information Systems and Business Organization. Addison-Wesley, pp. 160—162
Sanders D.: Computers in Business. MacGraw-Hill, pp. 319—323
Aron, ibid. pp. 106—112
Davis G.: Interduction to Computers. III Edition. McGraw-Hill, pp. 105—111

ad 3. jw.

ad 4. Nassi I., Shneiderman B.: Flowchart Techniques for Structured Programming. SIGPLAN Notices (ACM) vol. 8 (No. 8 — August 1973), pp. 12—26

ad 5. Cushing, ibid. pp. 261—263
Klerer M., Korn G. (Editors): Digital Computer User's Handbook. Section 1. McGraw-Hill, pp. 278—291

ad 6. Myers G.: Projektowanie niezawodnego oprogramowania, WNT 1980
Gilb T.: Software Metrics. Winthrop Publ., pp. 28, 36

ad 7. Davis, ibid. pp. 70—72, 141
Aron, ibid. pp. 35, 169—186
Sippl Ch. J.: Data Communications Dictionary. Van Nostrand Reindhold, New York 1976, p. 128

Najważniejszymi sposobami uruchamiania programów są:

— porównywanie każdego fragmentu programu (ang. code walk-through) z projektem
— bezkomputerowe przeglądanie (tzw. suche testowanie) logiki i składni całości napisanego programu (ang. desk check)
— sprawdzanie przez kompilację (ang. assembly check, compilation check)
— weryfikowanie poprawności działania programu względem danych próbnych (ang. program test using test data)
— korzystanie ze specjalnego oprogramowania diagnostycznego (ang. debugging procedures)
— niezależne sprawdzanie równoległych przebiegów komputerowych (ang. parallel runs) wielu kopii tego samego programu.

ad 8. Davis, ibid. pp. 142—146

Adaptowali:

A. B. EMPACHER
J. L. ROSSOWSKI

Nazwy własne produktów informatycznych

W artykułach, dokumentacjach i innych opracowaniach — oprócz terminów informatycznych, które skrupulatnie tłumaczymy na język polski korzystając ze słowników i z tej rubryki — pojawia się wiele różnych nazw oznaczających produkty sprzętowe lub programowe. Pochodzenie tych nazw jest często znane tylko nielicznym. Czasem stają się one nowymi rzeczownikami, jak np. termin angielski **spooling**, który według słownika oznacza r o l k o w a n i e, ale w słownictwie informatycznym jego rdzeń jest akronimem SPOOL (Simultaneous Peripheral Operations On Line), oznaczającym przechwytywanie operacji we-wy, wykonywanych współbieżnie z programem.

W informatyce nowe nazwy powstają wraz z pojawieniem się nowej firmy, komputera lub jego istotnej części, a także — każdego nowego elementu oprogramowania, takiego jak system operacyjny, język programowania i jego translator oraz różnego rodzaju programy pomocnicze. Ze względu na konieczność ochrony praw rynkowych, wszystkie te nazwy stają się bądź zastrzeżone przez firmę (copyright), bądź jej znakiem towarowym (trademark). W tabeli 1 wymieniono według dokumentacji firmowych nazwy zastrzeżone tylko przez dwie dość znane firmy komputerowe. Uwagę zwraca zastrzeżona postać tych nazw, określona dużym i małymi literatmi, a czasem też specyficzną pisownią (np. int$_e$l, TEX).

TABELA 1. PRZYKŁADOWY WYKAZ NAZW ZASTRZEŻONYCH

Digital Equipment Corporation

ASSIST-11	DECtape	OS/8
DIGITAL	DECsystem-10	PMA
DEC	DECSYSTEM-20	RTS-8
PDP	EDUSYSTEM	RSTS
DECUS	FLIP CHIP	RSX
UNIBUS	FOCAL	SBI
COMPUTER LABS	INDAC	TYPESET-8
COMTEX	LAB-8	TYPESET-11
DDT	LSI	TMS-11
DECCOMM	VAX	ITPS-10
DECnet	VMS	PDT
DATATRIEVE	MASSBUS	IAS
DIBOL	OMNIBUS	TRAX

Intel

BXP	Intelvision	Micromap
CREDIT	Intellec	Multibus
i	iOSP	Multimodule
ICE	iRMX	Plug-A-Bubble
iCS	iSBC	PROMPT
i_m	Library Manager	Promware
iMMX	MCS	RMX/80
Insite	Megachassis	System 2000
Intel	Micromainframe	UPI
intel		µSCOPE

oraz każda kombinacja elementów ICE, iCS, iRMX, iSBX, MCS lub RMX z liczbowymi przed- lub przyrostkami.

Wymogi praw ochrony znaku powodują, że każdy fragment tekstu w piśmiennictwie zachodnim, zawierający nazwy zastrzeżone, jest opatrzony coraz dłuższą listą właścicieli tych nazw. Przykład takiego tekstu w języku angielskim w formie żartu zamieszczono ostatnio w czasopiśmie „Communication of the ACM" (April 1984, Vol. 27/4.). W tabeli 2 pokazano zwykłe zdanie w języku polskim, wymagające dołączenia listy właścicieli użytych w nim nazw.

TABELA 2. ŻART Z MNOGOŚCIĄ NAZW ZASTRZEŻONYCH

Zwykłe zdanie w języku polskim:

Panna Ada [1] Pascal [2] z kompanem [3] Jerzym [4] pojechała nad Odrę [5] mikrobusem [6] o rejestracji VAX 780 [7], biorąc ze sobą jabłka [8] Macintosh [9] — i [10] jest to meritum [11] naszego oskarżenia"

wymaga zestawienia następujących odsyłaczy:

1) Ada jest znakiem towarowym zarejestrowanym przez rząd Stanów Zjednoczonych (Goverment, Ada Joint Program office)
2) Pascal jest nazwiskiem matematyka, Blaise'a Pascala, użytym przez Niklausa Wirtha z ETH w Zurichu.
3) Kompan jest spolszczoną nazwą mikrokomputera ComPAN zaprojektowanego przez Zakład Systemów Automatyki Komputerowej PAN w Gliwicach.
4) Jerzy jest spolszczoną nazwą systemu George zastrzeżoną przez International Computers Ltd.
5) Odra jest znakiem towarowym używanym przez zakłady Elwro we Wrocławiu.
6) Mikrobus jest lub będzie zastrzeżonym znakiem polskiego standardu magistrali dla systemów mikroprocesorowych.
7) VAX 780 jest zastrzeżonym znakiem towarowym firmy Digital Equipment Corporation.
8) Jabłko jest spolszczoną nazwą „Apple", będącą znakiem towarowym zastrzeżonym przez firmę Apple Computer Inc.
9) Macintosh jest znakiem towarowym zastrzeżonym przez firmę Apple Computer Inc.
10) Litera i jest nazwą zastrzeżoną przez firmę Intel Corp.
11) Meritum jest nazwą produktu zakładów ELZAB z Zabrza.

(Powyższe wyjaśnienia są podane z przymrużeniem oka).

W piśmiennictwie polskim nie przestrzega się tych zasad, trudno jest wyrokować, czy postępowanie takie narusza jakiekolwiek przepisy prawne... Może nowa ustawa o znakach towarowych rozstrzygnie ten problem.

Zostawiając na boku żarty, postaram się wyjaśnić pochodzenie niektórych nazw, sklasyfikować nazwy produktów informatycznych (ze względu na sposób powstawania) a następnie, zaproponować dla nich zasady pisowni i odmiany w języku polskim.

Dobra nazwa produktu jest częścią jego sukcesu u potencjalnych klientów. Autorzy starają się więc dobrać nazwy proste, samowyjaśniające się, a jednocześnie oryginalne. Wyróżnia się trzy klasy pochodzenia tych nazw: akronimy, kryptonimy oraz adaptacje.

Akronim jest skrótem utworzonym z pierwszych liter poszczególnych słów pełnej nazwy, z reguły charakteryzującej produkt.

Kryptonim jest nazwą utworzoną (np. przez skojarzenie) z części pełnej nazwy i z reguły ukrywającą tę właściwą nazwę.

Adaptacja nazwy polega na przejęciu nazwy własnej, już istniejącej, będącej najczęściej imieniem lub rzeczownikiem odnoszącym się do czegoś zupełnie innego. Czasem adaptacja ma na celu uhonorowanie czyjegoś imienia lub faktu.

Posługując się dalej tymi trzema klasami pochodzenia nazw, spróbuję przykładowo wyjaśnić najczęściej spotykane nazwy pochodzenia angielskiego lub amerykańskiego, w zakresie nazw systemów komputerowych i operacyjnych oraz języków programowania.

Nazwy systemów komputerowych są najczęściej identyfikowane z nazwami firm, w których one powstały. Akronimowa nazwa firmy przenosi się na nazwę produktu — mamy więc: **IBM, ICL, HP, CDC** (zbyt znane to firmy, aby należało tłumaczyć te skróty). Podobnie nazwy kryptonimowe otrzymują produkty: **Intel, Vax, Cyber** itp. W adaptacjach spotykamy nazwiska właścicieli firm: **Amdahl, Cray, Sinclair** oraz inne dowolnie dobrane nazwy, takie jak: **Apple** (jabłko) **Macintosh** (gatunek jabłka), **Eclipse** (zaćmienie), **Spectrum** (spektrum), **Commodore** (komandor) itp.

Nieco inna sytuacja istnieje w zakresie nazewnictwa systemów operacyjnych — podstawowego produktu programowego, związanego z systemem komputerowym czy firmą. Tutaj początkowo był wykorzystywany akronim **CP** (Control Program — program sterujący), a następnie **OS** (Operating System — czyli system operacyjny). Pojawiły się więc nazwy akronimowe systemów, takie jak:

PCP (Primitive CP) — pierwotny program sterujący

CP/M (CP/Monitor) — program sterujący dla mikrokomputerów

BOS (Basic OS) — podstawowy system operacyjny

TOS (Tape OS) — taśmowy system operacyjny

DOS (Disk OS) — dyskowy system operacyjny

RTOS (Real Time OS) — system operacyjny czasu rzeczywistego

MTOS (Multi Tasking OS) — wielozadaniowy system operacyjny oraz różne kombinacje w rodzaju: **RT, RST** (dla PDP-11), **RTDOS, RMOS, VMOS, PC-DOS, SOS** — uzupełniane numerami wersji, identyfikacjami typów komputerów itp. Z bardziej znanych skrótów możemy też odnotować:

VM (Virtual Memory) — pamięć wirtualna

VMS (Virtual Machine System) — system maszyn wirtualnych

MFT, MVT (Multiprogramming with Fixed/Variable Tasks) — wieloprogramowany ze stałą lub zmienną liczbą zadań

CTSS, TSS (Compatible Time Sharing System) — kompatybilny system z podziałem czasu

ISIS (Intel Systems Implementation Supervisor) — nadzorca implementacji systemów Intela.

Kryptonimy są wykorzystywane w mniejszym stopniu, chociaż i tutaj możemy odnotować:

Multics (MULTIprogramming Computer System) — system komputera wieloprogramowanego
Unix (UNIversal system X) — uniwersalny system X

Pozostaje jeszcze klasa adaptacji, w której spotykamy takie nazwy, jak: **George** (Jerzy — imię panującego), **Supervisor** (nadzorca — po polsku nie brzmi to zachęcająco), **Executive** (wykonawca) i wreszcie imiona z mitologii greckiej — **Kronos, Hydra** czy **Medusa.**

W nazwach języków programowania (a jest ich ponad 1000) najpełniej widać trzy klasy pochodzenia nazw, w większości tworzonych w oderwaniu od konkretnej firmy. W dziale akronimów posługujemy się po prostu akronimem **PL** (Programming Language — język programowania) otrzymując:
PL/I (PL One) — język programowania nr 1
PL/M, PL 65, PL 360 — dialekty PL/I
oraz wiele innych układów liter, uzupełniających podstawowy trzon PL dodatkowym określeniem, poczynając od: **APL** (A PL — pewien język programowania), **BCPL** (Basic biped PL — podstawowy kombinowany język programowania), **MPL** (Modular PL — modularny język programowania), **NPL** (Nonprocedural PL — nieproceduralny język programowania), **EL** (Extended L — język rozszerzony), **SL** (System L — język systemowy), **RTL** (Real Time L — język czasu rzeczywistego), aż do nazwy poetyckiej **PLITS** (PL In The Sky — język programowania w niebie).
W klasie tej odnotowujemy też inne akronimy, jak:
SR (Synchronizing Resources) — synchronizowanie zasobów
DP (Distributed Processes) — procesy rozproszone
CSP (Communicating Sequential Processes) — komunikowanie procesów sekwencyjnych
oraz
BASIC (Beginners Al-purpose Symbolic Instruction Code) — uniwersalny kod instrukcji symbolicznych dla początkujących

W klasie kryptonimów należy zwrócić uwagę na nazwy jednoliterowe np. znany język **C** oraz mniej znane **B, S** itp. Nazwy kryptonimowe języków powstawały nawet wcześniej niż akronimowe. W klasie tej występują nazwy tak popularnych języków, jak:
Lisp (LISt Processor) — procesor list
Fortran (FORmula TRANSlator) — tłumacz wyrażeń
Algol (ALGOri thmic Language) — język algorytmiczny
Cobol (COmmon Business Oriented Language) — język ukierunkowany na rozwiązywanie typowych problemów gospodarczych
Prolog (PROgramming LOGic) — programowanie logiczne
Forth (FouRTH generation language) — język czwartej generacji (ze względu na ograniczenie długości nazwy do pięciu znaków dla pierwszej implementacji języka usunięto literę „u"; forth — naprzód)
oraz oparte na skrócie — lan (LANguage) kryptonimy:
Modula (MODular LAnguage) — język modularny
Simula (SIMUlation LAnguage) — język symulacyjny
Loglan (LOGical LANguage) — język logiczny
Clan (Concurrent LANguage) — język współbieżny
Plan (Programming LANguage) — język programowania
Futlan (FUTure LANguage) — język przyszłości (propozycja).

Nazwy nowych języków są coraz częściej adaptacjami imion i nazwisk osób tworzących podstawy obliczeń mechanicznych i algorytmicznych. Mamy więc już języki:
Pascal — Blaise Pascal, 1623—1662, matematyk, twórca jednej z pierwszych maszyn do dodawania,
Ada — Ada Augusta Lovelace, 1815—1852, pierwsza programistka, córka George'a Byrona
Edison — Thomas Alva Edison, 1847—1931, słynny wynalazca, między innymi twórca podstaw techniki elektronowej.

W zapasie mamy takie nazwiska, jak: Babbage, Post, Turing, Neuman itp. oraz Anna — na cześć znanej mi programistki komputera MERA 400.

Istotnym uzupełnieniem nazw są dodatki rozróżniające poszczególne warianty produktu. Przykładowo, dla języków programowania wyróżniamy dodatki oznaczające:
— **wersje,** np. Fortran IV, Lisp 1.5
— **lata wprowadzenia,** np. Algol 68, Fortran 77
— **rozszerzenia,** np. Concurrent Pascal, Pascal Plus, Extended Basic
— **wersje implementacyjnie** związane z typem komputera lub firmy np. OMSI Pascal, UCSD Pascal, Pascal 1900, MBasic (Microsoft Basic) itp.

Przedstawiony spis nazw można wykorzystać w dalszych rozważaniach na temat sposobu zapisu i użycia nazw produktów informatycznych w języku polskim.

W języku angielskim nazwy będące akronimami i kryptonimami pisze się dużymi literami. Nazwy adaptowane zaczyna się dużą literą, pisząc je dalej przeważnie małymi — zależnie od redakcji tekstu. W języku polskim tylko akronimy powinny być pisane dużymi literami (bez kropek rozdzielających poszczególne znaki). Pozostałe nazwy, w myśl analogii z innymi nazwami własnymi, powinny być pisane małymi literami (jak np. fiat, okocim itp.). W praktyce jednak, nazwy własne zaczynają się dużą literą, czasem dodatkowo ujmując cały wyraz w cudzysłowy (np. „Fiat"). Często, w celu wyróżnienia nazw w tekście, pisze się je w całości dużymi literami (jak to przyjęto w tym czasopiśmie — na razie z wyjątkiem tego artykułu).

Reasumując, postuluję wszystkie nazwy własne produktów informatycznych pisać — w przypadku akronimów — dużymi literami, a pozostałe małymi, zaczynając od dużej litery. Należałoby jednak rozróżnić nazwy własne opisujące pewną grupę produktów tej samej klasy (np. język Fortran, Pascal) od konkretnych egzemplarzy danego produktu, będącego realizacją, implementacją związaną z określonym typem sprzętu lub z określoną firmę (np. FORTRAN 1900). W tym drugim przypadku powinniśmy uwzględnić pierwotną, wprowadzoną przez właściciela lub twórcę postać nazwy z dokładnością do wielkości znaków ich położenia oraz zastosowanych separatorów (np. dBase II, VisiCalc, TEX).

Drugim istotnym problemem językowym jest odmiana nazw produktów. Dla nazw pochodzenia akronimowego przyjmuje się brak odmiany, pozostawiając ją domyślności czytającego tekst (podobnie jak nie pisze się — w ONZcie lecz w ONZ). Tak więc, zdanie „Program zapisano w X dla Y działającego na Z" wymaga od czytającego wiedzy, że X jest nazwą języka Y — systemu operacyjnego, a Z — komputera. Lepiej byłoby zapisać zdanie w postaci „Program zapisano w języku X dla systemu operacyjnego Y, działającego na komputerze Z". Podobna sytuacja występuje w odniesieniu do pozostałych rodzajów nazw, przy czym nazwy te mogą być odmieniane przez przypadki, jeżeli nie są poprzedzone rzeczownikiem je objaśniającym, np. — w Fortranie, ale — w języku Fortran. W celu skrócenia zapisu tekstów i w przypadku wielokrotnego używania tej samej nazwy, postuluję unikać poprzedzania jej rzeczownikiem objaśniającym. Można więc mówić — w Basicu, w Fortranie, w Adzie, dla Pascala — odmieniając nazwy zgodnie z zasadami odmiany nazwisk pochodzenia angielskiego.

Dalszą konsekwencją przyswajania nazw produktów informatycznych jest tworzenie z nich rzeczowników ogólniejszego znaczenia i przymiotników. Trudno jest jednak kierować się tu jakimiś zasadami. Dla przykładu mówimy „jadę pekaesem", ale nie możemy powiedzieć „jadę pekapem, emzetką", chociaż zarówno PKS, jak i MZK, PKP są identycznie skonstruowanymi akronimami. Podobnie jest też w rozważanej przez nas tematyce. Możemy spokojnie napisać lub powiedzieć „program algolowy", „fortranowy" czy „procedura algolowa", „pascalowa", a nawet „adowa", ale z trudem zaakceptujemy zwroty — „program pielanowy" czy „forthowy", ze względu na duże uzależnienie od języka obcego. Myślę więc, że wchłanianie nazw własnych produktów informatycznych do polskiego słownictwa powinno opierać się na naturalnych zasadach, ale z unikaniem zbędnych żargonowych śmieci. A swoją drogą, patrząc na nasze informatyczne podwórko można zauważyć, że nazwę utworzyć jest łatwo, gorzej zaś z realizacją propozycji — a tym samym z popularyzacją i powszechną akceptacją samej nazwy.

WACŁAW ISZKOWSKI

Informatyka na Wiosennych Targach Lipskich

Wiosenne Targi Lipskie z racji terminu oraz swej wysokiej międzynarodowej pozycji, są każdego roku chronologicznie pierwszą prezentacją postępu w informatyce. W wyniku konsekwentnie przeprowadzanej w NRD na przestrzeni ostatnich kilkunastu lat integracji całokształtu działalności badawczej, przemysłowej, handlowej i usługowej pod znakiem firmowym kombinatu ROBOTRON, kraj ten osiągnął przodującą pozycję wśród krajów RWPG nie tylko dzięki szerokiemu asortymentowi i dobrej jakości sprzętu komputerowego oraz oprogramowania, ale również wszechstronnemu rozwojowi zastosowań informatyki praktycznie we wszystkich dziedzinach gospodarki i życia społecznego. Osiągnięty potencjał wytwórczy pozwala w pełni zaspokoić istniejące zapotrzebowanie rynku wewnętrznego, a ponadto przeznaczyć ok. 70% produkcji na eksport. Zgromadzone doświadczenia eksploatacyjne i szkoleniowe stanowią dodatkowy czynnik wspomagający ekspansję eksportową.

Odbiciem tej sytuacji jest powiększająca się z roku na rok, w zestawieniu z pozostałymi krajami RWPG, dominacja w Lipsku ekspozycji targowej NRD. Tegoroczna ekspozycja podkreślała tę dominację jeszcze bardziej, uwydatniając również zjawisko malejącego zainteresowania Targami ze strony potencjalnych konkurentów kombinatu ROBOTRON. Spadek zainteresowania osiągnął swe ekstremum wyrażające się w nieobecności producentów zachodnich, a nawet nowych, dynamicznych firm z krajów Trzeciego Świata (np. Tajwan), czego nie można tłumaczyć wyłącznie względami politycznymi (działania embargowe USA).

Relacja z tegorocznej ekspozycji targowej z konieczności ograniczy się więc głównie do prezentacji oferty NRD. Jest to dla nas o tyle korzystne, że zafascynowani mało dostępnym w polskiej sytuacji gospodarczej sprzętem światowej czołówki producentów, zmniejszyliśmy — w porównaniu z okresem sprzed kilku lat — zainteresowanie sprzętem produkcji naszego zachodniego sąsiada, co spowodowało, że obecnie bardzo mało o nim wiemy.

Ekspozycja NRD została tradycyjnie skoncentrowana w hali nr 15. Jej gospodarzami były kombinaty: ROBOTRON (kompleks informatyczny), NACHRICHTENELEKTRONIK (telekomunikacja), MIKROELEKTRONIK (układy scalone i urządzenia do ich produkcji) oraz AUTOMATISIERUNGSANLAGENBAU (automatyka przemysłowa). Podstawowa scenografia uwypuklała dwa główne obszary zastosowań sprzętu komputerowego: zarządzanie oraz wytwarzanie. Głównym szlagierem był 16-bitowy komputer osobisty robotron A 7100, oparty na uniwersalnym systemie modularnym MMS 16 (Mikrorechnermodulsystem). Dzięki zróżnicowaniu systemów operacyjnych oraz interfejsów, mikrokomputer ten może być stosowany we wszystkich dziedzinach zastosowań oraz w różnych rozwiązaniach organizacyjnych przetwarzania.

Podobne wymagania spełnia pokazywany już w roku ubiegłym, a obecnie zmodernizowany mikrokomputer 8-bitowy robotron 1715. Ma on pamięć RAM o pojemności 64 KB oraz możliwość przyłączenia do czterech jednostek pamięci na dyskach elastycznych 5,25" lub 8". W skład podstawowej konfiguracji wchodzą: przestawna klawiatura oraz ruchomy, monochromatyczny monitor ekranowy o pojemności 1024 lub 1920 znaków. Do mikrokomputera można przyłączyć różne modele drukarek: mozaikowych (robotron K 6311, K 6312, K 6313, K 6314 lub K 6316) oraz rozetkowych (robotron 1152/251 lub 1152/252), wyposażonych w różne rozwiązania mechanizmów do automatycznego wprowadzania i przesuwu papieru. Istnieje ponadto możliwość przyłączenia standardowej elektronicznej maszyny do pisania robotron S 6010, wyposażonej w interfejs V 24, a także nowego modelu drukarki robotron 1157/269, przystosowanej do wyprowadzania danych zarówno w postaci graficznej (szybkość 90 zn./s), jak i alfanumerycznej (szybkość 150 zn./s). Oprócz wymienionych drukarek kombinat oferuje również jako produkt OEM rozetkowe mechanizmy drukujące robotron S 8501, zapewniające wysoką jakość druku (tzw. jakość korespondencyjna — ang. letter quality) oraz zestawy znaków, specyficznych dla poszczególnych alfabetów narodowych.

Ze względu na bardzo szeroki zakres oraz efektywność zastosowań, szczególnie interesującą pozycją oferty NRD był ręczny czytnik pisma znormalizowanego OCR B. Urządzenie to o symbolu robotron K 6710 jest przeznaczone do automatycznego odczytu treści etykiet towarowych, zapisanych czcionkami pisma OCR B bezpośrednio na stanowiskach pracy w sklepach, magazynach czy produkcji.

Ofertę mikrokomputerową NRD uzupełnia mikrokomputer domowy robotron Z 9001. W połączeniu z telewizorem i magnetofonem kasetowym stanowi on tani sprzęt zaspokajający wymagania początkującego użytkownika oraz dydaktyki szkolnej.

Należy wreszcie wspomnieć o demonstrowanym już w roku 1984 systemie przetwarzania obrazów, robotron A 6472, przeznaczonym głównie do komputerowej interpretacji zdjęć meteorologicznych i kartograficznych Ziemi. Zastosowanie tego unikalnego w skali RWPG urządzenia jest coraz szersze (wykrywanie zasobów naturalnych, zanieczyszczeń środowiska, promieniowania cieplnego, ostrzeganie przed katastrofami żywiołowymi, diagnostyka w przemyśle, biologii i medycynie itp.).

Szczególnie duży nacisk w ekspozycji NRD położono na demonstrację sprzętu do projektowania i wytwarzania wspomaganego komputerem (CAD/CAM). Reprezentantem tego kierunku było stanowisko pracy konstruktora i technologa robotron A 6454 z nowym digitalizatorem (wejście foniczne), ploterem K 6418, drukarką graficzną 1157/269 i urządzeniem rastrującym K 8917.

Istotnym akcentem ekspozycji były również liczne przykłady zastosowań biurowych, zwłaszcza w dziedzinie bankowości, ubezpieczeń i obsługi informacyjnej. Stale doskonalony sprzęt (zwłaszcza monitory ekranowe) i coraz bogatsze oprogramowanie świadczą o konsekwentnych działaniach producenta w kierunku osiągnięcia poziomu wymagań współczesnych użytkowników.

W dziedzinie związanych z informatyką urządzeń telekomunikacyjnych zaprezentowano nowy dalekopis F 2000 produkcji firmy RFT. Stanowi on przełom w tej grupie sprzętu, ze względu na wprowadzenie sterowania mikroprocesorowego oraz dwóch bloków pamięci o łącznej pojemności 14 KB. Wzrosły również szybkość przesyłania (do 200 bodów) oraz drukowania (do 50 zn./s). Ta sama firma zaprezentowała także nowy modem VM 2400, umożliwiający transmisję z szybkością do 2400 bodów.

Imponująca była wreszcie ekspozycja kombinatu MIKROELEKTRONIK, pokazująca aktualne osiągnięcia NRD nie tylko w zakresie oferty elementów półprzewodnikowych, ale również urządzeń technologicznych do ich wytwarzania. Uzupełnieniem oferty tego producenta był specjalizowany komputer do gry w szachy, który pojawi się również w Polsce.

Szczupłość miejsca pozwoliła na bardzo skrótowe przedstawienie tylko najważniejszych pozycji oferty NRD. Pozycje te będziemy się starali w przyszłości dokładniej omówić. Z bardzo oszczędnej ekspozycji innych krajów RWPG należy wymienić głównie ZSRR, który m.in. zaprezentował jako nowości mikrokomputer ELEKTRONIKA 8020/3 oraz elektroniczną kasę OKA 301, a także ekspozycje CSRS, WRL i BRL, które jednak nie tylko nie pokazały nowości, ale również pełnej skali swych możliwości. Ta ostatnia uwaga odnosi się również do ekspozycji polskiej, która była w wyraźnej dysharmonii nawet z naszymi skromnymi osiągnięciami w dziedzinie sprzętu mikrokomputerowego.

WŁADYSŁAW KLEPACZ

Ogłoszenie • Ogłoszenie • Ogłoszenie

Oprogramowanie do COMMODORE C-64 piszemy na zlecenie instytucji, osób prywatnych. 27-400 Ostrowiec, skrytka pocztowa 40. EO/1103/K/85

Ogłoszenie • Ogłoszenie • Ogłoszenie

INSTYTUT ORGANIZACJI PRZEMYSŁU MASZYNOWEGO

poleca PORADNIK

Jak zawierać umowy w zakresie informatyki

Autorzy: B. Czachórska, J. Irlik, T. Mazurkiewicz, R. Pessel
Objętość 7 ark. aut., cena 700 zł

Poradnik zawiera obszerne omówienie podstawowych typów umów w zakresie informatyki oraz zagranicznych i krajowych doświadczeń z tej dziedziny. Stanowi pożyteczny zbiór informacji o umowach dotyczących sprzętu, oprogramowania, przetwarzania danych.

Jest jedynym w kraju źródłem tego rodzaju informacji przeznaczonych dla:

- informatyków świadczących usługi i zatrudnionych w działach przetwarzania danych,
- pracowników administracji, zlecających zadania i redagujących umowy,
- radców prawnych i innych służb, stykających się z nowymi sytuacjami organizacyjnymi i prawnymi, w wyniku rozwoju informatyki i jej zastosowań.

Z jego treści dowiesz się:

- Jak formułować klauzule prawne zabezpieczające interesy zleceniobiorców i zleceniodawców?
- Jakie są cele poszczególnych klauzul?
- Jak specyfika działalności i zastosowań komputerów wpływa na postanowienia umowne?
- Jak dbać o prawa autorskie?

Instytut Organizacji Przemysłu Maszynowego, Dział Wydawnictw.
00-879 Warszawa, ul. Żelazna 87,
tel. 20-48-31 wew. 310, teleks 813747 IOPM PL

EO/1351/K/85

ALMA-SERVICE

STUDENCKA SPÓŁDZIELNIA PRACY

w Katowicach

dysponuje kompletnymi, aktualnymi i realnymi informacjami o własnej działalności gospodarczej. Dane pochodzą z komputerowych systemów ewidencyjnych.

Poszukuje

instytucji lub osób z doświadczeniem w zakresie budowania i efektywnego wykorzystania modeli symulacyjnych, wymagających podejmowania decyzji. Celem współpracy jest zaprojektowanie odpowiedniego systemu informatycznego.

Nasze oczekiwania i warunki współpracy do omówienia w siedzibie Zarządu:

ul. Huberta 17a, 40-542 Katowice
tel. 518-045 w. 12, 30, 35, teleks 0315738

EO/1284/K/85

W skrócie • W skrócie

■ Jesienią ub.r. odbyła się pierwsza konferencja zorganizowana przez stowarzyszenie użytkowników systemu operacyjnego UNIX — European Unix Users' Group. Konferencja połączona z wystawą UNIX SYSTEMS 84 stanowiła okazję do oceny praktycznej przydatności tego najpopularniejszego dziś systemu operacyjnego. Uwagi dotyczyły głównie zastosowań systemu we wspomaganiu działalności firm handlowych. Wystawa zaprezentowała wiele nowości. Przeważały wersje UNIXA dla różnych typów procesorów (głównie firmy DEC — np. nowa wersja PDP-11) oraz systemy specjalizowane stworzone w oparciu o UNIX. Dominowała wersja UNIX V. Ciekawe były propozycje nowych rozwiązań sprzętowych — szczególną uwagę zwrócił 32-bitowy komputer ORION firmy HIGH LEVEL HARDWARE. Zajmuje się ona budową emulatorów dla prac nad językami sztucznej inteligencji. ORION służy do emulowania różnych typów maszyn, a wymiana mikroprogramów zapisywanych w procesorze segmentowym AMD2901C zapewnia sprawną realizację tej funkcji komputera. W wersji podstawowej ORIONA symulowana jest maszyna stosowa przeznaczona do współpracy z systemem UNIX 4.1 oraz językiem C. System UNIX jest m.in. podstawą systemów projektowania układów scalonych. Jednym z najnowszych produktów tego typu jest pakiet SCALD oparty na systemie UNIX 4.1C. Najmniejsze gabarytowo mikrokomputery pracujące w systemie UNIX zaprezentowała firma UNIQIX. Mikrokomputery te zadziwiają miniaturyzacją — w obudowie o wysokości ok. 6 cm mieszczą się — oprócz jednostki centralnej — dysk elastyczny oraz 10M-bajtowy dysk typu WINCHESTER. Z mikrokomputerem tym może współpracować jednocześnie trzech użytkowników, pod nadzorem wielozadaniowego systemu operacyjnego UNIPLUS, kompatybilnego z wielozadaniowymi wersjami systemu SPIRIT UNIX. (TW)

•

■ W ubiegłym roku pojawił się na rynku sprzętu elektronicznego nowy typ drukarki — magnetyczna bezdotykowa. Dokładna zasada działania drukarki nie jest jeszcze publikowana, znane są natomiast jej parametry użytkowe. Drukarka MP 6090 firmy BULL używa tzw. prostopadłej magnetycznej techniki zapisu, dającej dużą precyzję oraz możliwość zróżnicowanej prędkości oraz szerokości druku. Obsługa drukarki nie wymaga przygotowania specjalistycznego. Ma ona rozbudowaną diagnostykę uszkodzeń, obejmującą testy elektryczne i magnetyczne. Używając standardowego papieru, drukarka drukuje 90 stron (6 tys. linii) na minutę. Magnetyczna metoda druku zapewnia całkowitą odporność na zakłócenia elektrostatyczne, a wysoka jakość pisania utrzymywana jest bez powszechnie stosowanych systemów sterujących. Podstawowa wersja MP 6090 ma tzw. poziomą gęstość druku równą 10, 12, 15 znaków/cal, natomiast gęstość pionowa druku wynosi od 10 do 20 wierszy/cal w wersji rozszerzonej. Gęstość pozioma i pionowa może być zmieniana dowolnie, w trakcie druku. Drukarka zawiera cztery zbiory po 98 znaków graficznych. Zbiór ten może być znacznie rozszerzony — rozbudowa pamięci drukarki pozwala dysponować repertuarem 12 tys. znaków. Możliwe są ponadto zmiany kierunku druku, przez obrót osi kierunku druku o 0, 90, 180 i 270 stopni. (TW)

Mikołajczak B., Stokłosa J.: Złożoność algorytmów (1). Klasyfikacja problemów

INFORMATYKA 1985, nr 9, s. 1

Pierwsza część omówienia problemu złożoności algorytmów, zawierająca klasyfikację problemów oraz naświetlenie zagadnień związanych ze złożonością opisu struktury algorytmów i obliczeń programów napisanych w języku asemblera.

Bilski E., Fryźlewicz Z., Huzar Z., Kaliś A.: Rozległe sieci komputerowe

INFORMATYKA 1985, nr 9, s. 5

Charakterystyka rozwoju sieci komputerowych oraz szczegółowe omówienie modelu sieci ISO/OSI, prowadzącego do standaryzacji zasad współdziałania systemów komputerowych o różnej architekturze. W zakończeniu podano założenia przyjęte przy projektowaniu Międzyuczelnianej Sieci Komputerowej (MSK) w Polsce.

Waligórski S.: LOGO (3). Porównanie dialektów APPLE, SINCLAIR i COMMODORE 64

INFORMATYKA 1985, nr 9, s. 11

Zakończenie charakterystyki języka LOGO, zawierające szczegółowe porównanie wersji tego języka używanych na mikrokomputerach APPLE, SINCLAIR i COMMODORE 64.

Mikołajczak B., Stokłosa J.: Algorithm complexity (1). Classification of problems

INFORMATYKA 1985, No. 9, p. 1

First part of discussion on algorithm complexity problems, which includes classification of problems and presentation of questions connected with complexity of algorithm structure description and realization of programs written in assembler language.

Bilski E., Fryźlewicz Z., Huzar Z., Kaliś R.: Wide area networks

INFORMATYKA 1985, No. 9, p. 5

Characteristics of computer networks development and detailed discussion of the ISO/OSI network model, which tends towards standarization of cooperation principles of computer systems with different architecture. In the termination the assumptions for designing the MSK Intercollegiate Computer Network in Poland are presented.

Waligórski S.: LOGO (3). Comparison of APPLE, SINCLAIR and COMMODORE 64 language versions

INFORMATYKA 1985, No. 9, p. 11

Termination of the LOGO language characteristics, which includes detailed comparison of different language versions used on APPLE, SINCLAIR and COMMODORE 64 microcomputers.

Миколайчак Б., Стоклоса Я.: Сложность алгоритмов I). Классификация проблем

INFORMATYKA 1985, № 9, стр. 1

Первая часть обсуждения проблемы сложности алгоритмов, содержащая классификацию проблем и освещение вопросов связанных со сложностью описания структуры алгоритмов и вычислений программ написанных на языке ассемблера.

Бильски Э., Фрьезьлевич З., Хузар З., Кались А.: Глобальные вычислительные сети

INFORMATYKA 1985, № 9, стр. 5

Характеристика развития вычислительных сетей и подробное обсуждение модели сети ISO/OSI, ведущей к стандартизации правил взаимодействия вычислительных систем различной архитектуры. В заключение даны предположения принятые в проектировании межвузовской вычислительной сети в Польше.

Валигурски С.: ЛОГО (3). Сравнение диалектов APPLE, SINCLAIR и COMMODORE 64

INFORMATYKA 1985, № 9, стр. 11

Завершение характеристики языка LOGO содержащее подробное сравнение вариантов этого языка используемых на микро-ЭВМ APPLE, SINCLAIR и COMMODORE 64.

Mikołajczak B., Stokłosa J.: Algorithmenkompliziertheit (1). Problemenklassifikation

INFORMATYKA 1985, Nr. 9, S. 1

Erster Teil einer Besprechung von Algorithmenkompliziertheit, die Problemenklassifikation und Erläuterung von Kompliziertheit der Algorithmenstrukturbeschreibung und der in Assemblersprache geschriebenen Programme umfasst.

Bilski E., Fryźlewicz Z., Huzar Z., Kaliś A.: Ausgedehnte Rechnernetze

INFORMATYKA 1985, Nr. 9, S. 5

Eine Charakteristik der Rechnernetzeentwicklung und ausführliche Besprechung des ISO/OSI Netzmodells, das zur Standarisierung der Zusammenwirkung von Rechnersystemen mit verschiedener Architektur führt. Im Schlussteil wurden die Voraussetzungen zur Projektierung des MSK-Hochschulrechnernetzes in Polen angegeben.

Waligórski S.: LOGO (3). Ein Vergleich der Sprachedialekten von APPLE, SINCLAIR und COMMODORE 64

INFORMATYKA 1985, Nr. 9, S. 11

Beendigung von LOGO-Charakteristik, die einem ausführlichen Vergleich der auf Mikrorechner APPLE, SINCLAIR und COMMODORE 64 verwendeten Spracheversionen umfasst.

CSK—Computer Studio Kajkowscy

81-505 GDYNIA ORŁOWO, ul. Balladyny 3B, tel. 29-00-18

Komputer osobisty może być przydatny niemal na każdym stanowisku pracy. Wymaga jednak odpowiedniego oprogramowania użytkowego. W ramach tego oprogramowania oferujemy zainteresowanym dostawę uniwersalnych pakietów programowych:

BANK DANYCH CSK, TABPLAN CSK, TEKST CSK, TRANSCOM CSK

To doskonałe narzędzia pracy dla każdego. Aby z nich korzystać, nie trzeba być informatykiem! Zupełnie samodzielnie można tworzyć złożone systemy zarządzania przedsiębiorstwem, każdym przedsiębiorstwem; nawet najbardziej specyficzne uwarunkowania nie są przeszkodą.
To jednak jeszcze nie wszystko... Kiedy dotychczasowe problemy łatwo i szybko zostały rozwiązane — pojawiają się zupełnie nowe. Można wtedy bez kłopotów samemu udoskonalić dotychczasowy system!

BANK DANYCH CSK, TABPLAN CSK, TEKST CSK, TRANSCOM CSK

składają się w zakładowe systemy płacowe, osobowe, finansowo-księgowe lub magazynowe. Korzystając z nich, z łatwością można prowadzić planowanie, kalkulacje i sprawozdawczość. Można też sporządzać kosztorysy i oferty, a nawet prowadzić „automatyczną" korespondencję czy redagować dowolne teksty. Można wreszcie skorzystać z już zgromadzonych zasobów na komputerze ODRA (pod nadzorem systemu GEORGE-3), wykorzystując komputer osobisty jako inteligentny terminal — stację lub emulator TTY.

Nowość:

Oferujemy system operacyjny kompatybilny z CP/M 2.2. dla mikrokomputerów ROBOTRON 5120/5130 oraz systemy finansowo-księgowe FK dla dowolnych mikrokomputerów

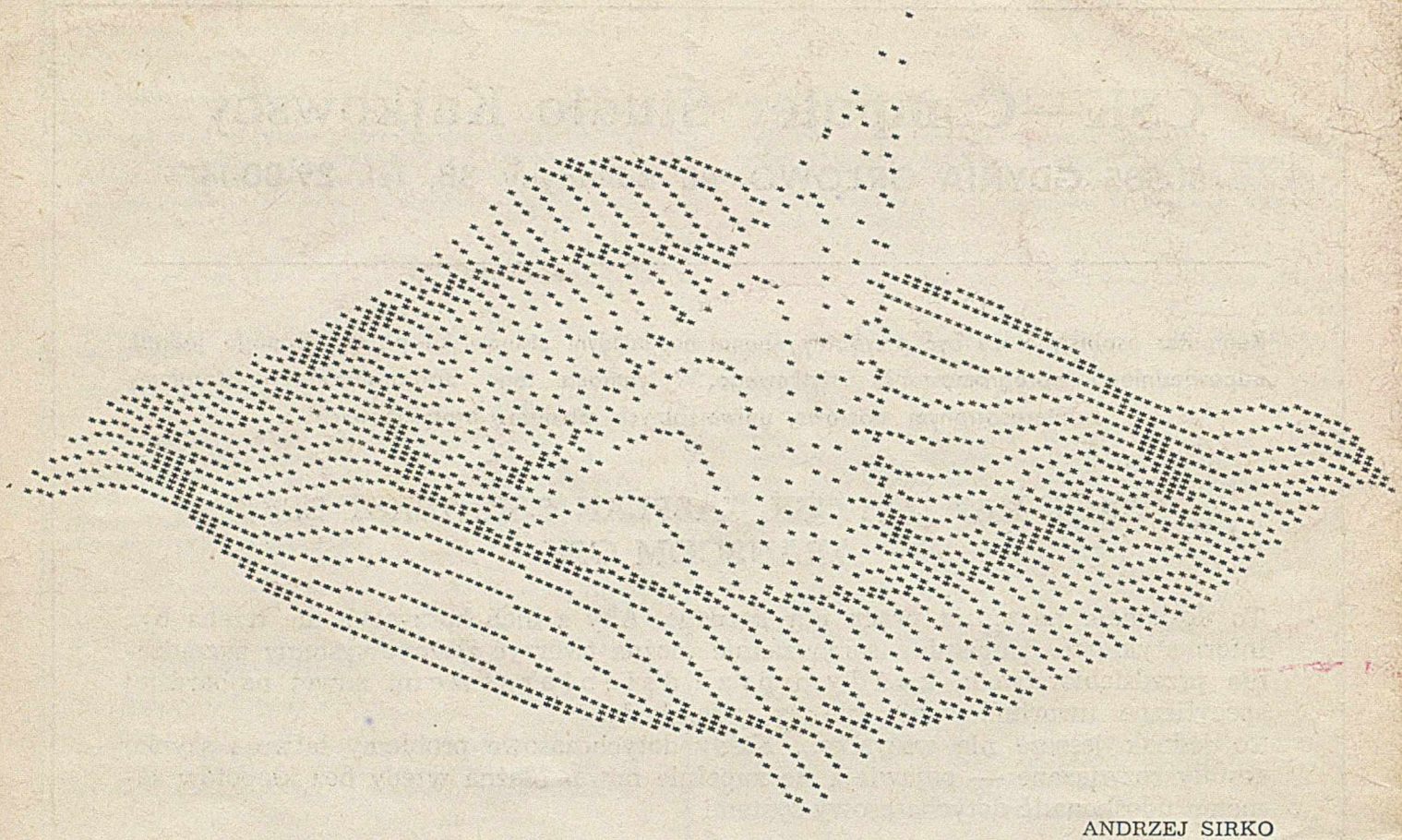
---

Szczegółowych informacji udziela:

CSK—Computer Studio Kajkowscy

81-505 GDYNIA ORŁOWO, ul. Balladyny 3B, tel. 29-00-18

EO/129/K/85



ANDRZEJ SIRKO
GRZEGORZ GANCARZ

WARUNKI PRENUMERATY NA 1986 R.

Prenumeratorzy zbiorowi — jednostki gospodarki uspołecznionej, instytucje i organizacje społeczne zamawiają prenumeratę dokonując wpłat na blankiecie „polecenie przelewu" rozszerzonym dla potrzeb Wydawnictwa o część dotyczącą zamówienia. Blankiety te będą dostarczane przez Zakład Kolportażu.

Prenumeratorzy indywidualni — osoby fizyczne zamawiają prenumeratę dokonując wpłaty w UPT lub NBP na blankiecie Wydawnictwa lub blankiecie NBP. Na odwrocie wszystkich odcinków blankietu należy wpisać tytuł czasopisma, okres prenumeraty, liczbę zamawianych egzemplarzy oraz wartość wpłaty.

Wpłacać należy na konto NBP III O/M Warszawa 1033-7420-139-11.

Prenumerata ulgowa — przysługuje wyłącznie osobom fizycznym — członkom SNT, studentom i uczniom szkół zawodowych. Warunkiem prenumeraty ulgowej jest poświadczenie blankietu wpłaty (przed jej dokonaniem) na wszystkich odcinkach pieczęcią Koła SNT, wyższej uczelni lub szkoły.

Sposób zamawiania prenumeraty taki sam jak dla prenumeraty indywidualnej.

Prenumerata ze zleceniem wysyłki za granicę — zamawia się tak jak prenumeratę indywidualną. Dodatkowo należy podać na blankiecie wpłaty nazwisko i dokładny adres odbiorcy. Cena prenumeraty ze zleceniem wysyłki za granicę jest dwukrotnie wyższa.

Przedpłaty na prenumeratę przyjmowane są w terminach:
— do 10 listopada na I kwartał, I półrocze i cały rok następny,
— do 28 lutego na II, III, IV kwartał i II półrocze,
— do 31 maja na III, IV kwartał i II półrocze,
— do 31 sierpnia na IV kwartał.

U w a g a!

Wpłaty na dwumiesięczniki przyjmowane są na okresy półroczne lub roczne.

Informacji o prenumeracie udziela — Zakład Kolportażu Wydawnictwa NOT-SIGMA, ul. Bartycka 20, 00-716 Warszawa, lub skr. poczt. 1004, 00-950 Warszawa, tel. 40-00-21 w. 249, 293, 297, 299 oraz 40-35-89 i 40-30-86.

Egzemplarze archiwalne czasopism — można nabyć za gotówkę w Klubie Prasy Technicznej w Warszawie ul. Mazowiecka 12, tel. 27-43-65 oraz w Dziale Handlowym Wydawnictwa ul. Bartycka 20 skr. poczt. 1004, 00-950 Warszawa, na rachunek dla instytucji lub za zaliczeniem pocztowym dla osób fizycznych.

Cena miesięcznika INFORMATYKA została ustalona na 120 zł za numer (35 zł — cena ulgowa).

Cena prenumeraty wg cennika					
kwartalna		półroczna		roczna	
normalna	ulgowa	normalna	ulgowa	normalna	ulgowa
360	105	720	210	1440	420